Data Analytics for Fault Prediction and Diagnosis in Wind Turbines

Kevin Leahy
BE (Hons)

Thesis submitted for the degree of Doctor of Philosophy

NATIONAL UNIVERSITY OF IRELAND, CORK

SCHOOL OF ENGINEERING

July 2018

Head of School: Prof Liam Marnane

Supervisor: Dr Dominic O'Sullivan

Contents

List of Figures

List of Tables

Acronyms

ANFIS Adaptive Neuro-Fuzzy Inference System.

ANN Artificial neural network.

AR Autoregressive.

CBM Condition-based maintenance.

CC Cluster Centroids.

CCFL Cluster Centre Fuzzy Logic.

CM Condition monitoring.

CMS Condition monitoring system.

CW Class Weight.

DA Deep autoencoder.

DARX Dynamic Autoregressive Exogenous.

DBSCAN Density-Based Spatial Clustering of Applications with Noise.

DD Direct Drive.

DFIG Doubly-fed induction generator.

DT Down time.

EAWE European Academy of Wind Energy.

EE EasyEnsemble.

ENN Edited Nearest Neighbours.

EWMA Exponentially Weighted Moving Average.

FRC Fully Rated Converter.

GWMA Generally Weighted Moving Average.

k-nn k-Nearest Neighbour.

LCOE Levelised cost of energy.

LDT Logistical delay time.

LR Logistic Regression.

MD Mahalanobis distance.

MTBF Mean time between failures.

MTTF Mean time to failure.

MTTR Mean time to repair.

NBM Normal behaviour modelling.

O&M operations and maintenance.

OEM Original equipment manufacturer.

PCA Principal Component Analysis.

PDF Probability Density Function.

PDM Predictive maintenance.

PF Potential Failure.

PH Prognostic horizon.

PLC Programmable logic controller.

PM Preventive maintenance.

PMG Permanent Magnet Generator.

PRC Partially Rated Converter.

RCM Reliability-centred maintenance.

RF Random Forest.

RMD Robust Mahalanobis Distance.

RO Research objective.

RUL Remaining useful life.

RUS Random Undersampling.

SCADA Supervisory control and data acquisition.

SMOTE Synthetic minority over-sampling technique.

SOM Self-organising map.

SVM Support vector machine.

TL TomekLinks.

Nomenclature

λ Failure Rate.

A Set of all alarms.

L_c^t Set of alarm instances with code c on turbine t.

a_c Alarm with code c.

ba Backup battery system faults.

bk Blade braking system faults.

fc Frequency Converter faults.

gb Gearbox faults.

gd Stoppages due to grid issues.

gn Generator faults.

l Specific alarm instance.

ma Stoppages due to maintenance or manual shut down.

mi Miscellaneous faults.

no Stoppages or curtailment as part of normal operation.

pt Pitch system faults.

rep Stoppages which resulted in repairs taking place.

sn Stoppages due to sensor errors.

t_e End time of a batch or alarm instance.

t_s Start time of a batch or alarm instance.

to Tower faults.

w_1, w_2 Windows of time in advance of a fault.

wa Stoppages due to weather issues.

yw Yaw system faults.

I, Kevin Leahy, certify that this thesis is my own work and has not been submitted for another degree at University College Cork or elsewhere.

Kevin Leahy

Abstract

As feed-in tariffs for wind energy are gradually being replaced by market driven auction-based systems, the need for cost savings at every stage of a wind energy project is more apparent than ever. A proven and effective way of reducing maintenance costs is through a condition-based maintenance (CBM) strategy. By using supervisory control and data acquisition (SCADA) system data instead of retrofitting a dedicated condition monitoring (CM) system, CM functionality can be gained at a fraction of the cost. This thesis investigates using SCADA system data for various levels of CM: fault detection, diagnosis and prediction.

First, a case study is presented on using classification techniques for CM using SCADA data. Various methods for dealing with the massive class imbalance seen in fault data are evaluated. It was found that all three levels of CM are possible using classification techniques, though with a high number of false positives. Adding a class-weight to the minority class or undersampling the majority class were found to be the best ways of dealing with class imbalance.

Sources of accurate failure data can be difficult to obtain for wind turbines. The second part of this thesis presents a novel way of building a historical failure database using alarm system and availability data. This was shown to produce an accurate database of unplanned stoppages related to assembly-level failures, scheduled maintenance, or grid, noise or shadow-related events.

Next, common issues with some of the classification approaches present in the literature are addressed, as well as the the lack of demonstration of how these approaches would perform in the field. A formalised framework with a prescribed list of steps following best practice guidelines is presented for performing CM using classification techniques on turbine SCADA data. A case study is performed which uses a sliding window metric to evaluate field performance, showing that such a system is effective at flagging faults in advance, but more data is needed to reduce the false positive rate.

It is noted throughout the thesis that turbine alarm systems have some consistent shortcomings, and do not live up to their full potential. Hence, a novel methodology is presented which uses clustering techniques to identify similar sequences of alarms as they occurred during unplanned stoppages. A case study applying the methodology showed that just under half of the 456 stoppages could be sorted into one of fifteen distinct types of alarm sequence.

This thesis is dedicated to role models I've looked up to all my life: my grandparents, Hannah and Maureen, and in particular my late granddads, John Leahy and John Ryan.

Acknowledgements

I'm very grateful to the following people and organisations, without whom this thesis would not have been possible:

To my supervisor, Dr. Dominic O'Sullivan, for the advice and mentorship he's given over the years. I'm very grateful for the feedback and encouragement I received at every stage, and the doors that are open to me because of it.

To Dr. Alice Agogino for very generously hosting me at UC Berkeley, and in particular for exposing me to the profound possibilities of development-focused projects. To Fulbright Ireland and SEAI, in particular Lisa Nic an Bhreithimh and Emma Loughney, without whom my unforgettable experience in the USA would not have been possible.

To all of my colleagues for the input and advice I've received: Colm, Ken, Brendan, Pete, all of the Seáns, John, Lily, Ioannis, Euiyoung and the OWP team. And to all of the other good friends I've made up in room 2.03 and beyond, for all of the laughs we've had over the years: Kieran, Ciarán, Paul, Lara, John, David, Dee, Deirdre and many others.

In addition, this thesis would not have been possible without funding from MaREI and SFI - to Angela and Gillian, thank you for your endless patience with all of my requests!

Finally, sincere thanks to my friends and family. In particular, my Mam and Dad, and my sisters, Rachel and Karen, for their support at every stage. And to Karla, for her endless patience, understanding and encouragement every step of the way.

"*Ní dhéanfaidh smaoineamh an treabhadh duit*"

Chapter 1

Introduction

1.1 Background and Motivation

1.1.1 The Need for Renewable Energy

Humanity has harnessed the power of renewable energy for millennia. In its most basic form, biomass such as wood or charcoal has been used for fire, while the wind was the primary source of energy for transport across seas and oceans. Later, both wind and water power were used to mechanically drive mills to refine grain and saw wood, or power pumps to move water and drain tracts of land. Furnaces and kilns for refining ore or producing bricks, breweries, and drying-houses all required heat energy sourced from biomass. In fact, much of the success of the Netherlands during the "Dutch Golden Age" of the seventeenth century can be attributed to the Dutch effectively making use of the abundance of peat and wind power available to them at the time (de Zeeuw 1978). As world powers raced through the industrial revolution and beyond, the vast amounts of heat required to power steam and create electricity meant that the world moved away from renewable sources to more energy dense fossil fuels. Global energy demand expanded at an exponential rate during the twentieth century, largely being met by coal, and later, oil and natural gas. These continue to dominate the world's primary energy supply today, as seen figure 1.1.

The burning of fossil fuels produces carbon dioxide (CO_2) and other "greenhouse gases" (GHGs). These GHGs occur naturally in small amounts in the atmosphere, and are vital for sustaining life. Through absorbing and emitting

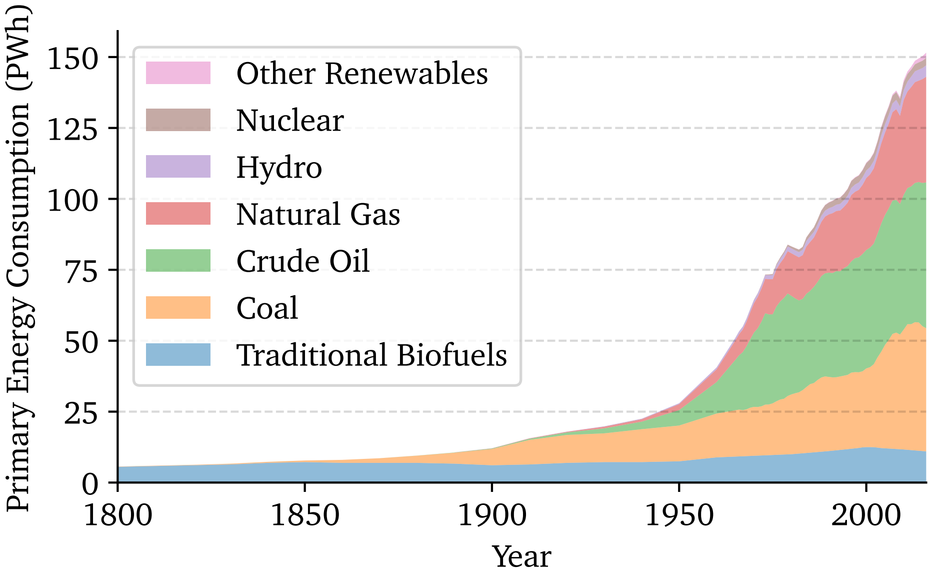


Figure 1.1: Global Primary Energy Use from 1800 to 2016 (Smil 2016, British Petroleum 2017)

thermal radiation (a phenomenon known as the "greenhouse effect"), they are important in sustaining a habitable temperature for the planet; it has been estimated that without them, the average surface temperature on Earth would be around -18°C (Schmidt et al. 2010). However, it is now known that the burning of fossil fuels on such an enormous scale has produced an over-abundance of these gases, and has amplified the greenhouse effect resulting in a rapidly warming global climate. Already, the global mean temperature has risen by about 1.2 °C since 1850 (Morice et al. 2012). This has had a high ecological impact, with shifts in the geographical ranges, abundance, migratory patterns and seasonal behaviours of many species being observed. More severe weather events have had a significant impact on the human population, and in many regions, crop yields have decreased, with those in poverty being particularly vulnerable to changes. These effects are expected to be greatly amplified over the coming decades, with extreme flooding, wildfires, food insecurity and civil unrest, and loss of entire ecosystems and livelihoods being expected should the status quo be maintained (IPCC 2014).

To mitigate these effects, there has been a concerted global effort to reduce the impact of CO_2 emissions in the atmosphere. Most recently, in December 2015, the Paris climate accord was signed by the governments of 195 different countries as part of an agreement within the United Nations Framework Convention on Climate Change. This agreement aims to respond to the threats imposed by man-made climate change by keeping a global temperature rise this century well below 2°C above pre-industrial levels, and to pursue further efforts to keep the rise below 1.5°C (United Nations Framework Convention on Climate Change 2015). These levels were identified as being realistically attainable while keeping temperatures below a threshold where the more severe effects of climate change become more apparent. Although this agreement is a very important moment in the fight against climate change, individual world and regional governments have been pursuing their own legislative efforts against climate change for some time.

1.1.2 Growth of Wind Energy

The EU's Renewable Energy Directive 2009/28/EC sets a binding target of 20% gross final consumption (GFC) of energy from renewable sources by 2020. As part of this, Ireland's target is 16% share of renewable energy in the national GFC by 2020. GFC includes the amount of energy consumed in transport and producing and distributing electricity and heat. The directive mandated that each Member State adopt a National Renewable Energy Action Plan (NREAP) to set out targets for the share of energy from renewable sources consumed in transport, electricity and heating by 2020. These sectoral targets are known as RES-E, RES-T and RES-H, respectively (European Parliament 2009). In Ireland in 2010, the share of energy-related CO_2 emissions from transport, electricity and thermal applications were 33%, 32% and 35%, respectively (Sustainable Energy Authority of Ireland 2017). To address this and meet the EU target, the NREAP for Ireland mandated targets of 40% RES-E, 10% RES-T and 12% RES-H (Department of Communications Climate Change and Environment 2010). To date, much of the RES-E target has come from wind energy, and this trend is expected to continue; in 2016, Ireland achieved a 27% share of renewable energy in its electricity generation mix, with 22% of this figure coming from wind. This reflected the year-on-year growth in installed capacity, with 400 MW installed in 2016 alone (Sustainable Energy Authority of Ireland 2017, Mc Garrigle et al. 2013). This growth in wind capacity mirrors the broader trends within the EU, and indeed the world, as seen in figure 1.2.

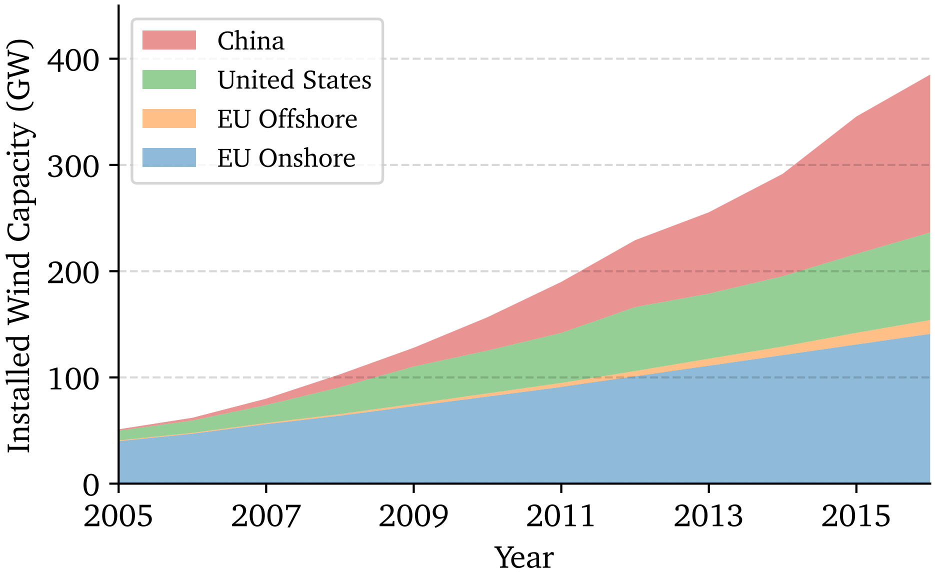


Figure 1.2: Installed wind energy capacity for EU, USA and China, 2005-2016 (EU has been split between on- and offshore) (British Petroleum 2017, WindEurope 2018)

Figure 1.2 also shows the annual growth in offshore wind in Europe. This is a relatively new industry, and the technologies underpinning it are beginning to mature. Offshore wind farms offer the advantages of decreased visual and environmental impact, as well as being able to capture the massive wind resource that is available out at sea. In fact, there is such a large resource potential offshore that WindEurope estimates it could provide between 80% and 180% of the EU's electricity demand at competitive prices by 2030. Ireland is in a very good position to capture some of this, with its Atlantic waters providing for a technical potential of 2600 TWh/yr by 2030 (BVG Associates & WindEurope 2017). However, in order to remain competitive, wind energy must remain an economically viable option for grid operators, investors, and other stakeholders.

1.1.3 Wind Energy Economics

The levelised cost of energy (LCOE) is a standardised measure used to compare the cost of power production from different generation technologies. It is the average total cost to build and operate a power-generating asset divided by the total energy output of that asset over its lifetime. It is usually used as a way of showing the average price of electricity that the asset must receive over its operating life in order for it to break even. Figure 1.3 shows the range of LCOE for solar, offshore wind and onshore wind globally for 2010 and 2016, with

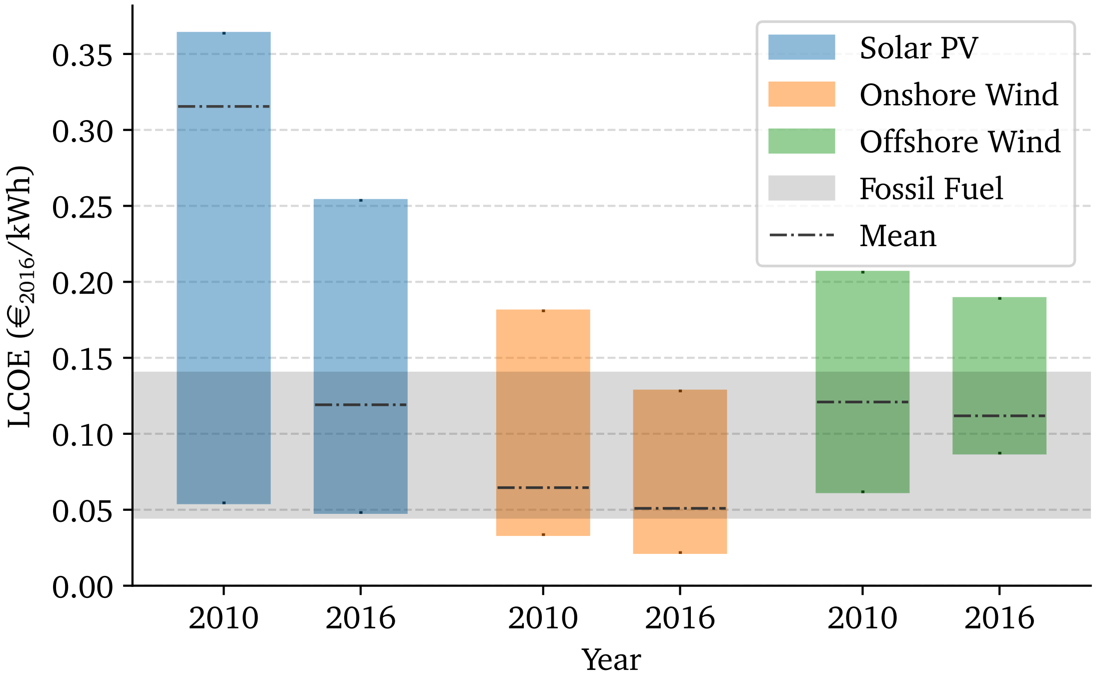


Figure 1.3: Ranges and averages of LCOE for solar PV, onshore wind and offshore wind compared to fossil fuels for 2010 and 2016 (International Renewable Energy Agency 2018)

the range of LCOE for fossil fuels given as a comparison. As can be seen, wind energy remains an attractive renewable electricity option, remaining cheaper than solar despite the latter's costs plummeting.

Although the price of wind energy has fallen in recent years, fossil fuels can still remain a cheaper option. In particular, the price of offshore wind is still much higher than onshore. Although offshore wind offers many advantages, the higher costs associated with constructing and operating a remote asset with limited access drive up the LCOE. Furthermore, potential future developments in fossil fuel generation or mining technologies mean that these may be more attractive to electricity providers in some markets. The cost of wind energy itself also has wide variations between different countries and markets (International Renewable Energy Agency 2018). Finally, wind power has traditionally been incentivised by a variety of support mechanisms. In Ireland, these have taken the form of the alternative energy requirement (AER) and renewable energy feed-in tariff (REFIT) (Foley et al. 2013). In many countries, these types of tariffs are starting to be replaced by market-driven auctions, driving up competition within the wind industry. Hence, lowering the LCOE of wind energy as much as possible, both on- and offshore, is important in order to maintain the growth of renewables in the energy mix.

The International Renewable Energy Agency has stated that operations and maintenance (O&M) costs for wind turbines account for a significant proportion of the overall LCOE, at 20-25%. This can extend up to 35% for turbines nearing the end of their life (Colone et al. 2017). The costs for onshore generally fall within the range of 20 to 40 €/kW/year, though this can fluctuate quite largely depending on the region and year; in Ireland in 2014 costs reached €70/kW/year. For offshore, costs are higher at roughly between €100/kW/year and €127/kW/year. Up to 80% of this cost, for both on- and offshore, is attributed directly to maintenance. It should also be noted that, given the rapid growth in deployment, wind turbine fleets are still relatively young. The German fleet, one of the oldest in the world, has an average age of just 10 years, while in China it is 5 years. The global average is just 6 years old. This can have implications for O&M costs as turbines begin to age - as more regular maintenance is needed, these costs may rise (International Renewable Energy Agency 2018, VGB PowerTech 2015).

The costs presented here (particularly for offshore) are in contrast to modern conventional fossil fuel generators which typically have lower O&M costs (excluding fuel). For example, closed cycle gas turbines (CCGTs) have maintenance costs of between 18 and 25 €/kW/year (VGB PowerTech 2015, U.S. Energy Information Administration 2016). While machines such as CCGTs operate in a constant, narrow range of speeds, wind turbines are designed to operate under a wide range of wind speeds and weather conditions. This means the stresses on components are comparatively higher, and leads to unique challenges in both designing and operating the turbines to achieve maximum possible availability while keeping O&M costs to a minimum. Corrective or unscheduled maintenance, whereby repairs are carried out when parts have become damaged, can be very expensive, particularly in the case of failures of major components; the cost of replacing a gearbox is in the region of €190,000, while a blade replacement is the region of €110,000 (Yang et al. 2014, Yürüşen et al. 2017). The fact that they are designed to operate autonomously and often deployed in remote locations means that daily visual inspections to ensure healthy operation are not normally feasible. For offshore turbines, this is compounded by the difficulties in getting access. While transport for a maintenance crew to visually inspect and repair a fault at an onshore turbine is relatively straightforward, offshore access requires expensive boat or helicopter transport which is dependent on sea and weather conditions (Peter Tavner 2012). Furthermore, if an unexpected fault *has* occurred, it is important that it has been correctly diagnosed in order to ensure the correct tools and parts have been brought along to avoid further trips.

1.1.4 Wind Turbine Maintenance Strategies

To avoid these unexpected failures and ensure proper component health, wind turbines have conventionally been serviced according to a preventive maintenance strategy, whereby periodic inspections are carried out at pre-planned intervals in order to assess the condition of various components and address any issues. This is coupled with a high-level alarm system which shuts down the turbine and alerts technicians when certain control variables such as temperatures or rotational speeds exceed their normal bounds. Typically, the length of time between these scheduled maintenance visits is decided by domain knowledge based on the historic failure rates of components. While this maintenance strategy avoids the long down time and replacement costs in the event of major component failure, it also means that in a lot of cases, a turbine has been brought off-line for inspection with minimal servicing work needed (Kothamasu et al. 2006). In an ideal scenario, there would be some way to continuously assess the condition of the turbine without bringing it off-line. In this scenario, the health of the turbine would be automatically monitored, and if it starts to deteriorate, information would be provided on if and when a specific type of maintenance action is required. This strategy is known as condition-based maintenance (CBM) (Duffuaa & Ben-Daya 2009).

In the wind industry, CBM has traditionally been carried out by installing additional vibration, oil-particulate, or other sensors to the turbine. The sensors can be expensive to install and record data at very high frequencies, meaning there are high bandwidth and storage requirements. Although these systems have seen significant success in the oil and gas industries, the cost justification for wind turbines is not nearly as strong (Yang et al. 2014). Furthermore, due to the fact that wind turbines operate at relatively low and variable speeds, the signals are harder to interpret and have not been as successful at fault prognosis as in other industries (Yang et al. 2013). However, being highly automated machines, there are a number of sensors already existing on the turbine used for its operation and control which can be leveraged to capture some of the functionality of a traditional CMS at a much lower cost. This data is typically aggregated and recorded at 10-minute intervals in the supervisory control and data acquisition (SCADA) system. Although at a comparatively lower resolution, some early successes have already been achieved (Tautz-Weinert & Watson 2017*c*). There are a number of different approaches which can be taken in attempting to do this.

1.1.5 SCADA-Based Condition Monitoring

A detailed review of all techniques will be discussed in chapter 2, however a brief overview will be given here. In its simplest form, simply tracking the performance of the turbine or monitoring certain values may show a change in turbine operation and, hence, be indicative of deteriorating component health (Wilkinson et al. 2014, Feng et al. 2011). An extension to this is to model the behaviour of the turbine under normal healthy operation, and build a residual which aims to show that deviation from this norm may be an indication that the health of some components are deteriorating (Schlechtingen & Ferreira Santos 2011, Butler et al. 2013). Classification-based techniques, on the other hand, attempt to train a machine learning classifier on sample data points from periods leading up to and during specific types of faults. The advantage of this is a single model may be trained to detect a broad range of incipient faults (Leahy, Hu, Konstantakopoulos, Spanos, Agogino & O'Sullivan 2018).

These classification-based approaches have seen success in domains such as credit card fraud detection, early detection of different types of cancers, and even fault detection in other industries (Hoadley et al. 2014, Abbasion et al. 2007, Ngai et al. 2011). However, much of the published research in the field of wind energy has yet to emulate these successes. Given the generally high availability of modern wind turbines, there exists an inherent difficulty in trying to predict faults against a backdrop of such a huge amount of fault-free data. Add to this the complexity of turbine operational states, where the turbine may be curtailed or shut down for any number of reasons, including grid issues, shadow or noise-related curtailment and low or very high wind speeds, and getting accurate classification results presents a significant challenge. Given this, many of the approaches detailed in the literature have limited scope and may not be reflective of the performance of the described systems in the field.

Accurate classification heavily relies on high quality training data - periods of faulty operation must be accurately labelled. Because there are so few fault samples, a small number of incorrectly labelled faults can heavily skew the results. Obtaining this accurate historical fault data can be problematic - fault records are often stored in unstructured formats by wind farm operators. Lack of conformity across operators, original equipment manufacturers (OEMs), or even individual maintenance technicians means that labelling training data for input into classifiers can be a tedious and ad-hoc process (Tautz-Weinert & Watson 2017*a*, van Kuik et al. 2016).

Turbine alarm systems can help identify different operational states, but even here there are issues. Alarms are triggered at a very high rate, particularly during fault events, meaning it can be unclear to operators what the root cause of the sequence of alarms was (Qiu et al. 2012). This also makes it difficult to use the alarm system data to retrospectively label historical operational data with specific reasons for downtime or curtailed operation, be it due to faults or other modes which fall within the bounds of normal operation (Gonzalez et al. 2016).

1.2 Research Objectives

As the wind energy industry moves towards a more proactive and predictive style of maintenance, leveraging existing SCADA and alarms data sources will be a key part of the solution. Classification-based approaches to fault prediction have yet to truly be utilised to their full potential, while alarm system data still largely relies on human interpretation by trained technicians armed with collective domain knowledge.

With this in mind, one of the objectives of this thesis is to investigate the level of fault prediction which can be achieved using classification techniques applied to SCADA data, and to see what specific techniques can be used to improve this performance. In order to achieve this, it will be necessary to evaluate whether alarm system data can be effectively used to build up training sets of past failures. With an understanding of the level to which the above can be achieved, it should be possible to formalise the most effective techniques into a generally applicable methodology and evaluate its performance when applied as a field deployed system. Furthermore, it will be investigated whether more contextual information can be gleaned for maintenance technicians from the high volume of alarms generated once a fault has actually occurred.

The main research objectives (ROs) of the thesis are then as follows:

RO1. Determine what level of condition monitoring can be performed using classification techniques

RO2. Investigate different techniques for dealing with classification based on imbalanced datasets and evaluate their suitability for fault detection, diagnosis and prediction

RO3. Determine whether information on historical failures can be accurately gleaned through analysis of the turbine alarms system, and whether this information can be used to create a complete and accurate training set for fault prediction

RO4. Design a comprehensive framework which incorporates all previous findings as well as best practices from literature and apply this methodology to evaluate its performance as a field-deployed system

RO5. Investigate whether the burden of analysis on maintenance technicians during fault events can be effectively reduced by gleaning information from the high volume of generated alarms

1.3 Thesis Structure

The rest of this thesis is laid out as follows:

- **Chapter 1** gives an introduction to the broad area of research and frames the motivation for undertaking this work
- **Chapter 2** provides a comprehensive background to the field of wind turbine maintenance, including maintenance theory, component reliabilities and current maintenance practices and the data they rely on. A comprehensive overview of wind turbine fault detection, diagnostics and prognostics from SCADA and alarm system data is also provided, including the techniques available to use and the benefits, challenges and theory involved in applying them.
- **Chapter 3** presents a case-study in detecting, diagnosing and predicting a number of different turbine faults. Within this, an investigation into various techniques used to address issues associated with the imbalanced nature of fault data sets is presented.

- **Chapter 4** outlines many of the issues researchers face in accurately labelling historical operational data with periods of faulty or fault-free operation. A method is presented which applies rules to alarm system data to address these issues and build an accurately labelled database.
- **Chapter 5** presents a formalised and prescriptive framework for wind turbine fault prediction using classification techniques. It uses the system of labelling training data described in 4, and describes the development of an alert system which notifies operators of impending faults and realistically simulates the deployment of this in the field using a new set of data.
- **Chapter 6** presents a methodology for grouping together similar patterns of alarm sequences which appear during fault events through the use of clustering techniques. This is done with a view to help operators give context and diagnosis options to future faults.
- **Chapter 7** summarises all of the findings and research output in this thesis. It presents the main conclusions drawn from the research, and the natural extensions and future research areas identified from it.

1.4 Research Output

The following journal publications have directly arisen out of work contained within this thesis:

- K. Leahy, R. L. Hu, I. C. Konstantakopoulos, C. J. Spanos, A. M. Agogino, and D. T. J. O'Sullivan, "Diagnosing and Predicting Wind Turbine Faults from SCADA Data Using Support Vector Machines," International Journal of Prognostics and Health Management, vol. 9, no. 1, pp. 1–11, 2018.
- K. Leahy, C. Gallagher, P. O'Donovan, and D. T. J. O'Sullivan, "Cluster analysis of wind turbine alarms for characterising and classifying stoppages," IET Renewable Power Generation, vol. 12, no. 10, pp. 1146–1154, Jul. 2018.
- K. Leahy, C. Gallagher, P. O'Donovan, K. Bruton, and D. T. J. O'Sullivan, "A Robust Prescriptive Framework and Performance Metric for Diagnosing and Predicting Wind Turbine Faults based on SCADA and Alarms Data with Case Study," Energies, vol. 11, no. 7, pp. 1–21, 2018.

In addition, the following conference publications have directly arisen from the thesis:

- K. Leahy, R. L. Hu, I. C. Konstantakopoulos, C. J. Spanos, and A. M. Agogino, "Diagnosing Wind Turbine Faults using Machine Learning Techniques Applied to Operational Data," 2016 IEEE International Conference on Prognostics and Health Management, pp. 1–8, 2016.
- K. Leahy, C. Gallagher, K. Bruton, P. O'Donovan, and D. T. J. O'Dullivan, "Automatically Identifying and Predicting Unplanned Wind Turbine Stoppages Using SCADA and Alarms System Data: Case Study and Results," in Journal of Physics: Conference Series, 2017, vol. 926, no. 1

Additionally, the following publication represents work which draws on some of the content of this thesis:

- R. L. Hu, K. Leahy, I. C. Konstantakopoulos, D. M. Auslander, C. J. Spanos, and A. M. Agogino, "Using Domain Knowledge Features for Wind Turbine Diagnostics," in 2016 15th IEEE International Conference on Machine Learning and Applications (ICMLA), 2016, pp. 300–307.

Although not directly related, the author additionally contributed to the following publications while undertaking the work for this thesis:

- C. V. Gallagher, K. Leahy, P. O'Donovan, K. Bruton, and D. T. J. O'Sullivan, "Development and application of a machine learning supported methodology for measurement and verification (M&V) 2.0," Energy and Buildings, vol. 167, pp. 8–22, 2018.
- C. V. Gallagher, K. Bruton, K. Leahy, and D. T. J. O'Sullivan, "The suitability of machine learning to minimise uncertainty in the measurement and verification of energy savings," Energy and Buildings, vol. 158, 2018.
- P. O'Donovan, C. Gallagher, K. Leahy, S. Blake, K. Bruton, and D. T. J. O'Sullivan, "A systematic mapping of industrial cyber-physical systems research for Industry 4.0," International Manufacturing Conference, 2017.
- P. O'Donovan, K. Leahy, K. Bruton, and D. T. J. O'Sullivan, "An industrial big data pipeline for data-driven analytics maintenance applications in large-scale smart manufacturing facilities," Journal of Big Data, vol. 2, no. 1, p. 25, Dec. 2015.

- P. O. Donovan, K. Leahy, D. Ó. Cusack, K. Bruton, and D. T. J. O'Sullivan, "A data pipeline for PHM data-driven analytics in large-scale smart manufacturing facilities," in Annual Conference of the Prognostics and Health Management Society 2015, 2015, vol. 6, pp. 1–10.
- P. O'Donovan, K. Leahy, K. Bruton, and D. T. J. O'Sullivan, "Big data in manufacturing: a systematic mapping study," Journal of Big Data, vol. 2, no. 1, p. 20, Dec. 2015.
- K. Leahy, K. Bruton, and D. T. J. O' Sullivan, "Implementing the Green Batch: A Case Study: Continuous Statistical Evaluation to Achieve the Most Energy Efficient and Reliable Process," in 19th IEEE International Conference on Emerging Technologies and Factory Automation, 2014.

1.5 Novel Contributions

This section outlines the specific novel contributions of this thesis.

Chapter 3 (Leahy et al. 2016, Leahy, Hu, Konstantakopoulos, Spanos, Agogino & O'Sullivan 2018):

- It was found that all three levels of CM (fault detection, diagnosis and prognosis) are possible using classification techniques, though with a high number of false positives.
- The addition of a class weight to the minority class(es), or simply undersampling the majority class, are the best way of dealing with class imbalance in the training stage

Chapter 4 (Leahy et al. 2017):

- A novel methodology for building an accurate historical failure database from alarm system and availability data is presented. The methodology shows assembly-level failure rates with accurate time stamps, as well as stoppages related to noise or shadow related curtailment, grid issues and planned maintenance.

Chapter 5 (Leahy, Gallagher, O'Donovan, Bruton & O'Sullivan 2018):

- A novel framework is presented which describes a list of steps, using best practice guidelines, for performing condition monitoring using classification techniques on wind turbine SCADA data.

- A further case study is presented which demonstrates the efficacy of such a system as it would perform in the field, showing the value of using a sliding window metric

Chapter 6 (Leahy, Gallagher, O'Donovan & O'Sullivan 2018):

- A novel methodology is presented which uses clustering techniques to identify similar sequences of alarms as they occurred during unplanned stoppages
- A case study applying the methodology showed that just under half of the 456 stoppages could be sorted into one of fifteen distinct types of alarm sequence

Chapter 2

Background

This chapter is intended to give the reader an overview of the research area and give context to the place this thesis takes within the broader literature. As part of this, the background and theory to many of the concepts and techniques used in later chapters of the thesis is given. Section 2.1 will give an overview of maintenance theory, including different maintenance strategies and the motivation for each. Section 2.2 will discuss some of the data sources which can be leveraged on wind turbines for both maintenance, monitoring and reliability reporting. Sections 2.3 to 2.6 will give an overview of wind turbine reliability and failure rates, what kinds of faults occur on wind turbines, and their impact on generating costs and down time from an onshore and offshore perspective. Section 2.7 will give a broad and high-level overview of the prevailing maintenance strategies in use in industry, drawing on some of the theory from section 2.1. Finally, section 2.8 will review current research into the techniques used in implementing some of these strategies; specifically, condition monitoring techniques based on wind turbine SCADA data.

2.1 A Background to Maintenance Theory

High reliability is an important aspect in all industrial assets and equipment - fewer breakdowns mean fewer breaks in operation, mean higher revenue and less expenses related to repairs and replacements. Reliability is generally defined as a measure of how seldom an item fails or needs repair; it is the probability that a system will perform its intended function over a specified period of time (International Electrotechnical Commission (IEC) 2010). Generally, the objective of any well-planned maintenance strategy is to minimise the amount of down time and lost production due to faults, as well as any repair costs that these faults may incur. With this in mind, here are some definitions that will be helpful throughout the rest of this chapter:

- Fault/failure: These terms generally mean that something has happened to a system so that it can no longer function as intended. In parts of the literature, fault refers to instances where a system has been brought off-line because of a *suspected* failure, whereas failure refers to when actual damage has taken place. In other parts, the terms are used interchangeably. In this work, unless otherwise stated, these terms are used interchangeably to mean either or both of these things, where the meaning will be clear from the context.
- Mean time to failure (MTTF): The mean time from when a system is repaired/fully operational to when it next fails. This is a measure of how much time a system can operate for before failure, independent of the time it takes to repair that system.
- Mean time to repair (MTTR): The mean time it takes from beginning a repair operation to when the system is fully operational, excluding any delays due to logistics (see below)
- Logistic delay time (LDT): The time it takes from when the system faults until the repair operation begins (due to repair crew travelling to site, waiting on parts, etc.). This also includes any additional logistical delays that may arise throughout the repair operation.
- Down Time (DT): The total amount of time the system is off-line due to failure, i.e. $\text{DT} = \text{LDT} + \text{MTTR}$
- Mean time between failures (MTBF): The mean expected time from the beginning of one failure to the beginning of another, i.e. $\text{MTBF} = \text{DT} + \text{MTTF}$
- Failure rate (λ): This generally refers to the number of failures per system or piece of equipment over a certain period of time. In this way, it can be thought of as $\lambda = 1/\text{MTBF}$

Note that failures in any system are not usually normally distributed over time (Hameed et al. 2010), so these are high-level metrics used to give a rough indication of system maintenance requirements. Although systems are generally designed to have high reliability, even with clever design and engineering choices all equipment eventually deteriorates over time from being exposed to the various loads and stresses of its operating environment (Duffuaa & Ben-Daya 2009). With this in mind, there are a number of avenues to pursue in terms of a maintenance strategy for an asset. Figure 2.1 shows an overview of various options available to maintenance managers when deciding on an appropriate strategy. These will be discussed in the following sub sections.

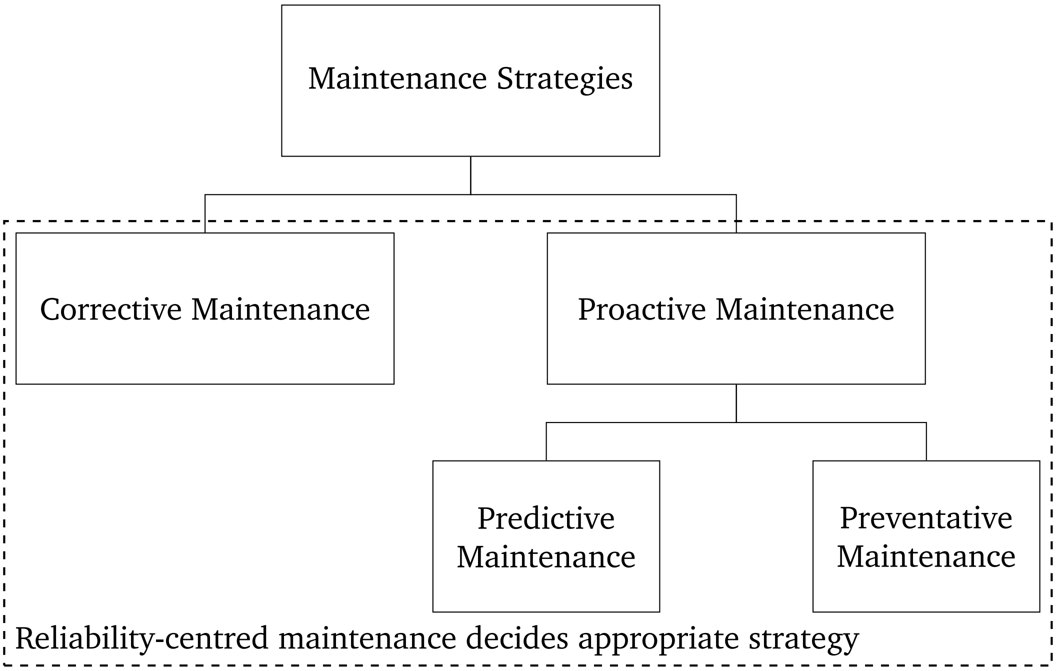


Figure 2.1: An overview of various maintenance strategies

2.1.1 Corrective Maintenance

The oldest and most basic maintenance strategy is corrective maintenance, or run-to-failure. This approach essentially boils down to "fix it when it breaks", and can be attractive as it requires very little planning or analysis. It is appropriate in applications where failures are very rare and repair costs are minimal (Kothamasu et al. 2006, Susto et al. 2015). However, running until failure means there is no control for the operator in terms of when or how an asset goes off-line due to component malfunction. The unscheduled nature of the downtime means that the repairs may need to be done at an inconvenient time, or at times when production yields would otherwise have been high; in the case of wind turbines, during rated wind speeds. Because it was not planned, spares may not be readily available which can lead to increased LDT. Furthermore, because component health is not monitored or inspected, a small fault can quickly develop into a catastrophic failure. For example, a problem with the emergency batteries in a wind turbine's pitch system can lead to safety mechanisms failing during periods of grid failure, leaving the turbine un-controllable in high wind speeds. This can turn what was originally a small repair into a potentially dangerous catastrophic failure. Hence, although corrective maintenance represents the least expensive maintenance strategy to implement and operate, repair costs of running to failure far outweigh the benefits in most cases (Butler 2012). Therefore there is a strong motivation to move from a *corrective* maintenance strategy to a *proactive* one, whereby maintenance is performed before a problem arises.

2.1.2 Reliability-Centred Maintenance

This motivation to move to a more efficient maintenance strategy is what led to a framework known as reliability-centred maintenance (RCM), originally developed in the latter part of the 20th century to optimise aircraft maintenance schedules (Nowlan & Heap 1978). The modern standard is contained in SAE JA-1011 (SAE 2009). RCM can be thought of as a prescribed list of steps that must be taken to evaluate the best type of maintenance strategy for each component or assembly in a system. It breaks an asset down into various subsystems, assemblies and components from a standardised, OEM-agnostic taxonomy specific to that type of asset. It then identifies lists of possible failure modes and mechanisms and possible root causes for each part of this taxonomy. Failure mode here describes the type of failure and its location within the system, the mechanism describes the way in which it failed, while the root cause describes the underlying reason for the failure (Peter Tavner 2012). In other words, the failure mode describes *what* failed, the failure mechanism describes *how* it occurred, while the root cause describes *why* it failed. An example of this is shown in figure 2.2. Here, the failure mode is main shaft failure. Possible mechanisms are through fracture or deformation, with the root causes of each of these failure mechanisms given below, e.g. high cycle fatigue or corrosion for shaft fracture.

Once this information has been collected, RCM decides on a suitable maintenance strategy for each component based on a prescribed list of evaluation steps. As can be seen in Figure 2.1, RCM makes a distinction between two

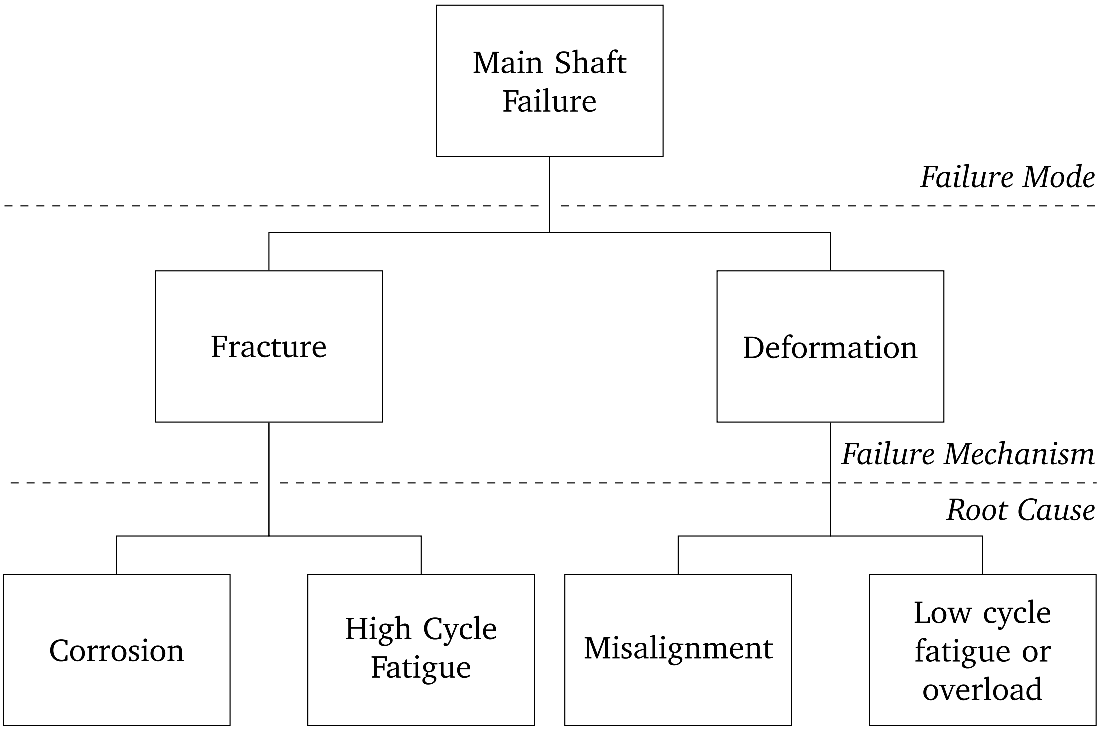


Figure 2.2: Relationship between a failure mode and its possible mechanisms and causes for a wind turbine shaft. This example taken from (Peter Tavner 2012)

broad types of maintenance strategy - corrective maintenance, as detailed previously, and proactive maintenance. Proactive maintenance is so-called as it aims to proactively service an asset to avoid unscheduled downtime. This further breaks down into *preventive maintenance* (PM) and *predictive maintenance* (PdM), described below.

2.1.3 Preventive Maintenance

PM represents the most straightforward proactive approach, so-called as it involves servicing an asset in order to prevent unscheduled downtime. Maintenance is performed on a periodic basis at a pre-determined interval, usually estimated from the historic distribution of λ for various components. The interval is selected so as to maximise the predicted MTBFs while minimising the likelihood of unplanned downtime.

This can be a very efficient strategy for components that wear out in a very predictable and repeatable pattern. However, for components that degrade in a stochastic way based on a number of random uncontrolled variables, maintenance tasks can be performed more frequently than absolutely necessary. While this reduces the severity of failures and maintenance tasks compared to corrective maintenance strategies, the component life may not be fully exhausted when maintenance occurs, meaning that in a majority of cases the asset has been brought off-line unnecessarily (García Márquez et al. 2012). Figure 2.3 shows this clearly, where PM leads to twice as many servicings as an optimal condition-based predictive strategy (discussed below).

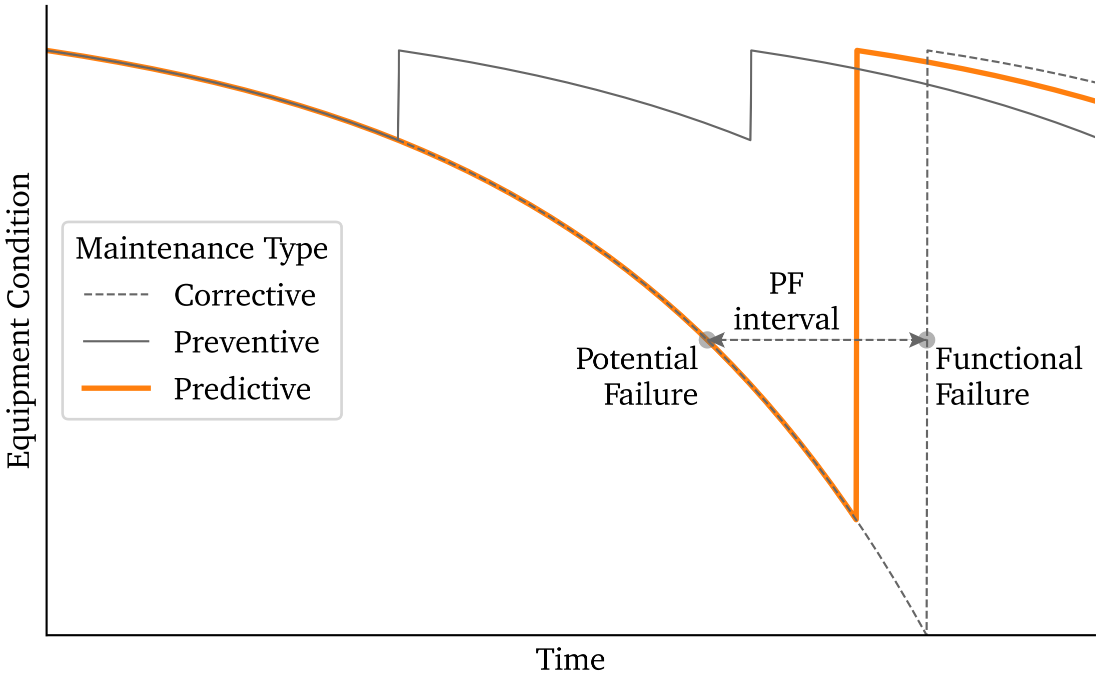


Figure 2.3: Comparison of the frequency with which machinery must be brought off-line for different maintenance types. Also demonstrated is the PF-interval, showing equipment condition deteriorating through the point where it can be noticeably detected (the potential failure) to when the functional failure occurs

2.1.4 Predictive Maintenance and Condition Monitoring Systems

PdM aims to increase maintenance efficiency by adaptively determining when maintenance is to be carried out. Maintenance is performed only when necessary, most commonly through the use of condition monitoring (CM) techniques on various components. The use of CM techniques to implement a PdM strategy is also known as condition-based maintenance (CBM). CM involves continuously monitoring the health or condition of a piece of equipment, and detecting incipient failures within some window before they occur, known as the *PF-interval* (Kandukuri et al. 2016). This interval describes equipment condition continuously deteriorating through a point known as *potential failure* all the way to *functional failure*. Potential failure is defined as an indication that a functional failure is in the process of occurring or is about to occur, while functional failure is defined as the point at which the system condition has deteriorated to where it is not able to carry out its function at a desired level of performance, i.e. the fault has occurred. The condition indicator used varies widely depending on the domain, but examples include particle counts in lubricating oil, or the measured error of some process variable from known acceptable bounds of normal operation. Figure 2.3 shows this clearly, whereby a potential failure is detected using CM, and maintenance is carried out. This figure also shows the benefits of using such a strategy compared to PM and corrective maintenance.

2.1.4.1 CMS Functionality

Although the ultimate goal of CBM is to detect a potential failure within the PF-interval with enough time to spare for maintenance to be carried out before functional failure occurs, CM systems (CMSs) generally have varying levels of functionality. These levels are broadly and loosely defined depending on context, but generally fall into one of the following categories: *fault detection*, *fault diagnosis* and *fault prognosis*.

Fault detection is the simplest of these, monitoring whether something has gone wrong in a system or not. At a whole-system level, this can be as straightforward as communicating whether or not a piece of equipment is operational. It also includes instances where a fault has occurred in a component, but the system continues to operate - in these instances fault detection is important so that the operator may shut down the asset before further damage occurs in related components or assemblies.

Fault diagnosis, meanwhile, communicates *what* has gone wrong in the system - the location of the fault at a sub-system or component level is established. Hence, a suitable maintenance task may be chosen and information about the fault relayed to the maintenance team. Fault diagnosis may also be able to determine the failure mechanism or root cause of the fault, providing even more valuable information so that appropriate action can be taken.

Prognostics, or predictive diagnostics, is the ultimate goal of CBM. It deals with predicting faults before they occur so that maintenance decisions can be taken ahead of time. It aims to detect a potential failure, and, if possible, provide operators with an estimate of the remaining useful life (RUL) for a component, as well as a confidence estimate for how accurate this prediction is. The prognostic horizon (PH) is a metric used to evaluate a prognostic technique, defined as the maximum time in advance of failure for which the RUL estimate for a particular technique has shown to be within acceptable bounds of accuracy. An example of this is shown in figure 2.4, where two algorithms' historical RUL predictions are evaluated against a ground truth. Here, algorithm 1 has a PH of 40 days, while algorithm 2 has a PH of 30 days.

An accurate RUL estimate can directly help with maintenance planning and logistics - of particular importance to offshore maintenance. However, it is also a much more complex procedure and inherently contains uncertainty (Saxena et al. 2008). Even with prognostics functionality, fault detection and diagnostics are still important - if prognostics fails for whatever reason, then the posterior event analysis is still necessary.

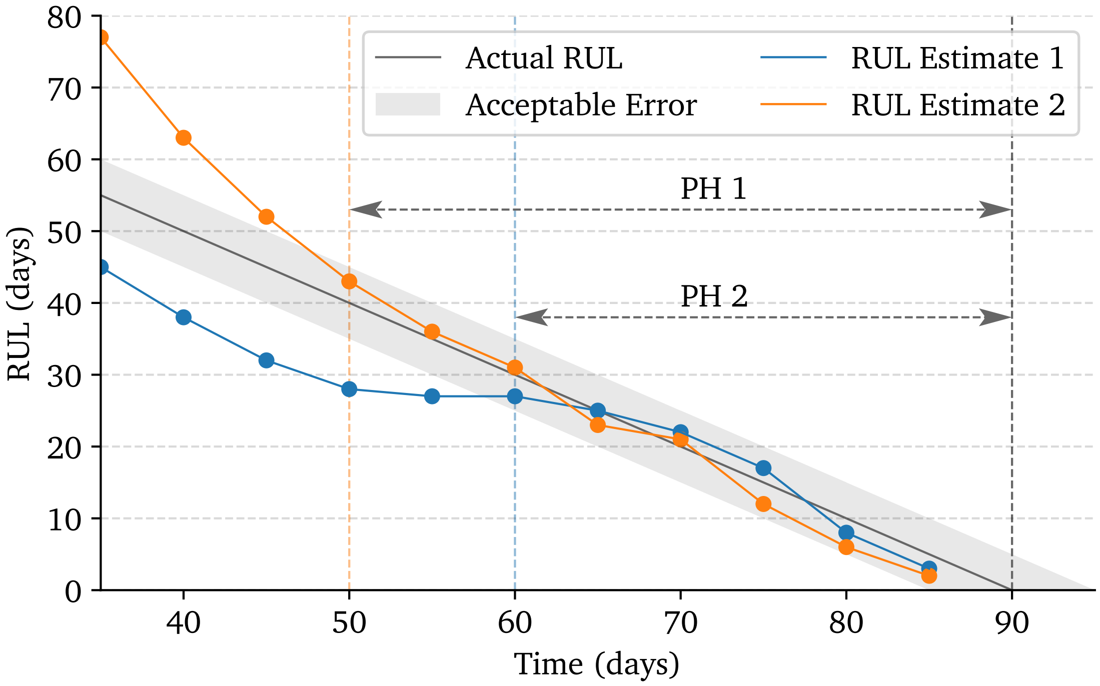


Figure 2.4: PH of two different CM techniques

2.2 Data Sources

Modern wind turbines have a number of sources of data which are used to monitor the turbine in terms of performance, availability and fault detection. These range in function from simple maintenance logs generated by repair technicians, to highly granular CMS signals. This section is intended to give a high-level overview of their functionality relevant to turbine maintenance strategies.

2.2.1 SCADA Data

The turbine's control system, usually centred around an industrial programmable logic controller (PLC), ensures that the turbine operates in a safe, stable and efficient manner. This controller can be interfaced through the turbine's SCADA system. SCADA systems originated in the oil and gas industry to provide an input/output communication system for various sensors and actuators. This meant that processes could be altered on-the-fly, and live information about system status could be communicated. In wind turbines, the majority of SCADA points provide information about the system rather than acting as manual inputs, with the exception of some manual overrides for safety, testing and maintenance (Peter Tavner 2012). This data is used to give operators and OEMs a high-level overview of performance at a turbine-, farm-, or fleet-level. In recent years, however, leveraging this data for more advanced CM has become an ongoing area of research, and will be discussed extensively in section 2.8.

SCADA data is typically split into three parts: operational data (usually recorded at 10-minute intervals), availability data (commonly recorded at 10-minute intervals to match the SCADA data), and alarms data (recorded instantaneously when alarms or information or warning messages are generated). Note that in much of the literature the term "SCADA data" refers specifically to the 10-minute operational data, with alarms and availability data used for the other parts. This will also be the case in this thesis, with the caveat that where it is not clear from the context, the specific type of SCADA data will be explicitly stated.

2.2.1.1 Operational Data

SCADA operational data is usually recorded continuously at 10-minute intervals (Tautz-Weinert & Watson 2017*c*). This can take the form of the average, minimum, maximum or standard deviation of live values recorded by the controller in the previous 10-minute period (Leahy, Hu, Konstantakopoulos, Spanos, Agogino & O'Sullivan 2018). Signals such as the turbine power output, wind speed,

Table 2.1: 10 Minute Operational Data

TimeStamp	Wind Speed (avg.) m/s	Wind Speed (max.) m/s	Wind Speed (min.) m/s	Power (avg.) kW	Ambient Temp (avg.) °C	Bearing Temp (avg.) °C
09/06/2014 14:10:00	5.8	7.4	4.1	367	17.9	25.0
09/06/2014 14:20:00	5.7	7.1	4.1	378	17.5	24.6
09/06/2014 14:30:00	5.6	6.5	4.5	384	17.6	25.1
09/06/2014 14:40:00	5.8	7.5	3.9	426	18.1	23.7

temperatures of various components, electrical signals and environmental conditions such as anemometer-measured wind speed and ambient temperature can be recorded. Different turbine manufacturers record different signals, and the systems can vary a lot from OEM to OEM in terms of the number and type of points recorded, and in some cases the resolution of the data. A sample of some typical turbine SCADA data is provided in table 2.1.

2.2.1.2 Availability Data

Availability is a measure of the total time a system is operational and available to perform its function as intended. In the field of wind energy, it has a specific meaning and, as will be explained in section 2.3, forms an important part of a turbine's warranty or service contract. Various OEMs report availability data in different ways. It can usually be accessed in some capacity by the SCADA system and communicates the proportion of time the turbine was operating in a particular availability "category" in a given period, e.g. in 10-minute time stamps to match the SCADA operational data. These categories cover broad situations where the turbine is generating, or available but not generating due to weather or grid-side events, or unavailable due to repairs or scheduled maintenance.

2.2.1.3 Alarm System Data

Turbine control systems include a number of sensors to monitor certain operating parameters such as temperatures, speeds, fluid levels, voltages, etc., and ensure the turbine is operating correctly. The controller attempts to keep the turbine within acceptable bounds of operation for many of these parameters, and warning or fault alarms and information messages (sometimes collectively

Table 2.2: Sample alarm system data from a given day

t_s	t_e	Code	Description	Category	Severity
02:13:38	07:56:14	a_{41}	Normal Operation	No Fault	Information
07:56:14	08:37:32	a_{91}	Low wind cut out	Weather	Information
08:37:32	23:44:02	a_{22}	Normal Operation	No Fault	Information

referred to as "status messages") are generated to provide information to operators about the current turbine state. Turbine alarm systems vary between manufacturers but OEMs generally attribute at least three different levels of severity to these alarms and messages:

- *Information messages* are generally to communicate changes in certain operating conditions, e.g. when the wind speed is too low for generation, a change in grid conditions, or a manual switch has been engaged.
- *Warning alarms*, on the other hand, are generated when the control system detects operating conditions or control variables that veer close to the limits of certain acceptable bounds
- *Fault alarms* are generated when these thresholds are exceeded

Information and warning alarms normally do not have much of a direct effect on turbine operation, although information alarms can be used to communicate that turbine production has been curtailed. Fault alarms generally cause the turbine to shut down, and some kind of intervention, as detailed in section 2.5, is needed before it comes back on-line. Instances of individual alarms when they are triggered have the following characteristics: start time, t_s, end time, t_e, code and description. Additionally, most alarm systems have some variation of an OEM-assigned category and severity. The severity refers to whether the alarm is an information, warning or fault alarm.

Alarms and messages are generated with instantaneous time-stamps, and cover many aspects of turbine operation. They usually give a description of where in the turbine the alarm originated, and the severity of the alarm (e.g. warning message, fault, information message, etc.). They can be used for limited CM, as will be discussed in section 2.7. There are also issues with the volume of alarms generated by turbines, and these will be detailed, along with a proposed solution, in chapter 6. A sample of turbine alarm system data is provided in table 2.2.

2.2.2 Maintenance Logs and Work Orders

Different operators, owners, OEMs and maintenance contractors record maintenance call-outs and repair activity in different ways. These can range from hand-written forms which describe any works carried out in an unstructured format, to detailed digital work orders input to a maintenance management system where work completed and materials consumed can be selected from a prescribed list. Depending on the quality of the data, these maintenance logs can be used to root-cause faults, and provide reliability information for components.

2.2.3 CMS Data

The term CMS when used in the context of the wind industry almost always refers to dedicated third-party CMSs which are independent of the turbine SCADA and alarms data. These originally appeared on turbines in the 1990s under pressure from certification bodies and are usually fitted to specific high-value components, such as the main bearing, gearbox or blades. They consist of vibration, oil particulate, or electrical sensors and strain gauges, and measure signals at a very high frequency, typically in the kHz range.

The CMS data stream is separate to the SCADA data, and in many cases is operated by the third party CMS supplier due to the sophisticated analyses required in interpreting signals. Some SCADA systems interface with the CMS so that alarms generated by the CMS sensors may also show up in the alarm system, but apart from this the two systems are largely separate entities providing different levels of CM. Although similar CMSs applied to traditional rotating machines such as steam turbines or aircraft engines are ubiquitous in their respective industries, the same successes have not yet been emulated in the wind industry, as will be discussed in section 2.7.2.

2.3 Reliability vs. Availability

As stated in section 2.1, reliability is the probability that a system or component will work as intended over a specified period of time. This is different to availability, which, in the field of wind energy, generally refers to the amount of time that a turbine is *available* to produce power from a technical standpoint, should the grid and wind conditions be within its design specification. This means that times when the turbine was not operating due to low or high wind speed, or noise-, grid- or shadow-related issues are still counted as "available", while times when it was down due to faults, repairs or routine maintenance/upgrades, are counted as "unavailable". Production-based availability is now gaining traction within the industry, where availability is measured in terms of missed energy production rather than missed generating-time (Peter Tavner 2012, Conroy et al. 2011). The different operating categories used to calculate availability can be operator-specific, or conform to standards such as IEC TS 6400-26-1 for time-based availability, or IEC TS 6400-26-2 for production-based availability (International Electrotechnical Commission (IEC) 2010).

A certain contractual availability is usually guaranteed by the OEM for a certain period at the start of a wind farm's operation, known as the *warranty period*. The exact specifics of this contract, including the equation used to calculate contractual availability and the period that the warranty covers, will vary from project to project according to owner and operator requirements. This "contractual" availability could also have "carve outs" so that specific allowances of repair-related or any other type of down-time still count as available (DNV GL 2017). In this thesis, availability refers to "technical" availability, where no such extra allowances are counted.

While turbine reliability depends on the construction of the turbine's components and is predictable (similar components will have similar failure rates, as will be seen), availability depends not only on reliability, but also on turbine maintenance strategy and location/access logistics (Spinato et al. 2009). Hence, there is a strong need to optimise maintenance strategy to maximise availability.

Onshore wind turbines have in recent years become highly reliable machines, with availabilities of upwards of 98%. This high availability is in part due to the PM strategies which have been adopted by OEMs over the years, though in many cases there is scope to raise this figure even higher by adopting more advanced maintenance strategies (van Kuik et al. 2016). Furthermore, as will be discussed in section 2.6, these strategies do not lend themselves as well to offshore installations (Ribrant & Bertling 2007, Peter Tavner 2012), where availabilities are much lower, in the region of 84% - 93% (Maples et al. 2013, Stehly et al. 2016).

2.4 Wind Turbine Taxonomies

As described in Section 2.1.2, RCM requires that a standardised taxonomy be built for an asset, though building a taxonomy is good practice for effective maintenance in general, and not just specific to RCM. Taxonomy in this instance refers to how parts of the turbine are broken down into their various assemblies and component parts. Broadly, this means applying a common OEM-agnostic nomenclature to major parts such as the generator or gearbox, and the individual components that make up these systems. One of the most obvious advantages of employing a system like this is that it allows operators to view and compare the failure rates and reliabilities of the assemblies of different turbine models within their fleet. The major assemblies of a wind turbine are shown in figure 2.5.

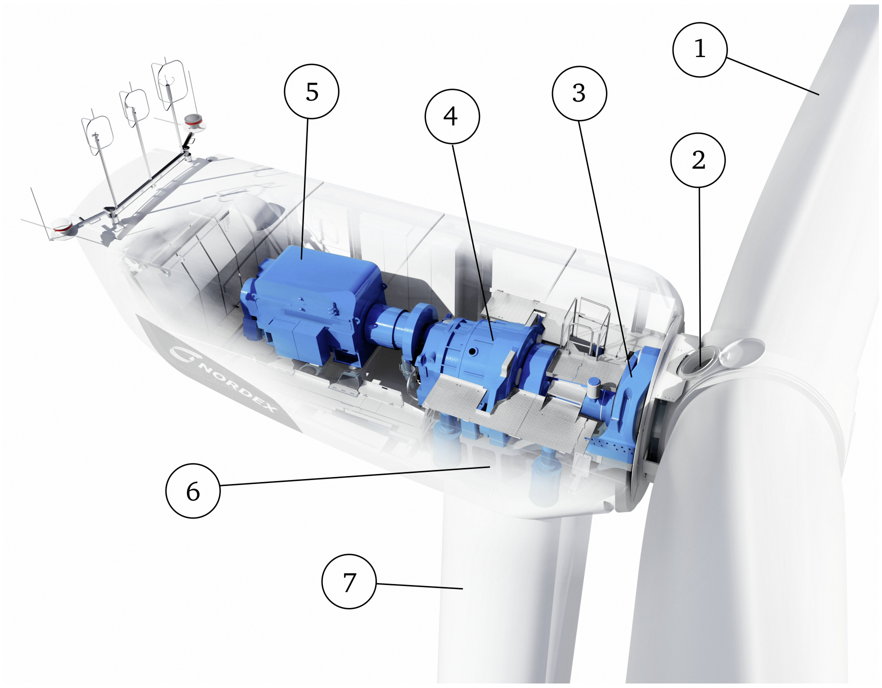


Figure 2.5: Structure of a modern wind turbine, showing major assemblies: (1) Blades; (2) hub and pitch mechanism; (3) main bearing; (4) gearbox; (5) generator; (6) yaw drive; (7) tower. Image courtesy of Nordex SE

In comparison to conventional fossil fuel generation technologies, standardised taxonomies for wind energy were developed relatively recently. Examples include a taxonomy developed by Sandia Laboratories for collecting reliability information on the US fleet (Hill et al. 2009), and another developed by VTT for similar purposes in Finland (Stenberg 2011). Two of the most widely used taxonomies are the Reference Designation System for Power Plants (RDS-PP) standard and the ReliaWind taxonomy. The RDS-PP taxonomy, developed by VGB PowerTech e.V. (VGB PowerTech 2014), was directly adapted from similar power industry taxonomies and designed to be consistent with designations of other power system types. The EU FP7-funded ReliaWind project developed its own recommended taxonomy (Wilkinson et al. 2011). This taxonomy adopts the following nomenclature:

- System, i.e. the wind turbine
- Sub-system, e.g. rotor & blades
- Assembly, e.g. pitch system within the rotor & blades module
- Sub-Assembly, e.g. pitch drive within the pitch system
- Component, e.g. a motor or bearing within the pitch drive

This taxonomy is openly available. However, it was based on turbines which were manufactured before 2008. Turbine designs have changed dramatically since then, with larger, higher capacity turbines now the norm (U.S. Department of Energy Office of Energy Efficiency and Renewable Energy 2017, WindEurope 2018). Reder et. al address this in their reliability study by proposing a slightly updated version of the ReliaWind taxonomy which is better suited to more modern turbines (Reder et al. 2016). The Reder et. al taxonomy will be used in this thesis for referencing turbine parts and faults, and is found in full in Appendix A.

2.5 Definition of Failure

As will be seen, serious failures in wind turbines are relatively rare and usually concern a major component such as the blades, generator or gearbox. As Tavner states, more often than not, unplanned stoppages are initiated by the controller after detecting an operating condition that is outside of normal/safe bounds, rather than due to serious component malfunction (Peter Tavner 2012). This can be due to an over-temperature, over-speed, current surge, or similar, and causes the controller to disconnect the turbine from the grid, put the blades into emergency feather condition and bring the turbine to a full stop. Once the turbine has been shut down in this way, one of the following must happen for it to be brought back on-line (in order of severity):

1. an automatic restart initiated by the controller;
2. a manual restart initiated manually from a remote control centre;
3. a manual restart initiated by an on-site technician after a field visit; or,
4. a repair operation initiated after a field visit followed by an on-site manual restart

These stoppages make up the vast majority of wind turbine faults, so that the "failure rates" discussed in the literature can be better thought of as "unplanned stoppage rates". However, in order to distinguish between simple controller-initiated automatic resets and more serious faults where there was some physical repair needed in order to get the turbine running again, various reliability studies apply different further criteria within this definition. The specific criteria used largely depend on the type and quality of data available. This data largely comes from SCADA alarm and operational data, work orders, and summary data from O&M reports. These data sources are largely discrete and in many cases are not designed to be mapped to a standardised taxonomy for reliability analysis, which leads to issues with the treatment of non-uniform sources of data (Wilkinson et al. 2011, van Kuik et al. 2016, Reder et al. 2016, Tautz-Weinert & Watson 2017*a*). This is an issue that will be discussed at length in chapter 4. Significant data processing is required, and, depending on the data available, different studies tackle this in different ways. Because of this, different studies have different definitions of what constitutes a "failure" in the data.

2.6 Wind Turbine Reliability and Failure Rates

Wind turbines are highly autonomous robotic devices, so they are designed to be highly reliable. However, unlike the oil and gas industry, which has standard ISO 14224:2016, there is no universal maintenance or reliability reporting standard for the wind industry (ISO 2016). This, along with the commercial sensitivity of such data, meant that for a long time there were few publicly available sources of reliability data for wind turbines, so the actual failure rates of various assemblies and components were difficult to obtain. As a result, the knowledge of component reliability was fragmentary and sometimes anecdotal, even among operators and professionals in the industry (Spinato 2008). However, there are now a number of different studies spanning different time periods, turbine designs, and turbine ages.

Although different reliability studies use different definitions of failure, and concern different types of turbines, it is still possible to get a general understanding of the assemblies or components with the highest failure rates and down time. This is important in order to understand the motivation for improving maintenance strategies, and see where these strategies can be targeted. It should be noted here that failure rate in most cases refers to failures/turbine/year. With that in mind, a number of different reliability studies are discussed in this section.

2.6.1 Early Reliability Studies

One of the earliest data sources for reliability studies was the WMEP database in Germany. This covered a period of 15 years from 1989-2006, with a database of just under 1,500 turbines (Faulstich et al. 2008). Analysis of this database is found in (Hahn et al. 2007, Faulstich et al. 2011). Meanwhile, the WindStats databases, comprising of German (WSDK) and Danish (WSD) turbines, and the LWK database, also containing German turbines, are analysed in (Spinato et al. 2009). This combined set of data had over 5200 turbines covering a period of 13 years from 1993-2006. A similar analysis performed on 750 Swedish turbines from 1997-2005 is found in (Ribrant & Bertling 2007).

These studies largely relied on manually entered information from standardised reports which were recorded after each routine maintenance or repair visit. In the studies of WMEP, LWK and WindStats databases, the data came from paper and digital forms which were manually filled out by maintenance technicians. This information was voluntarily supplied by turbine operators and maintenance contractors, and failures were defined as instances where an unscheduled maintenance call-out to the site was required, i.e. after an on-site manual restart, or repair followed by a restart (Spinato et al. 2009, Faulstich et al. 2011). The Swedish database also used a manual reporting format collected in a similar manner, and in the analysis a failure is defined as when a repair or replacement has been carried out on a turbine (Ribrant & Bertling 2007). In this way, the Swedish study used a narrower definition of failure.

The number of turbines in these studies and the long period of time over which they occurred make them very valuable - the trends which can be drawn on turbines which have been operating for more than 10 years is important as data on turbines which are nearing the end of their supposed 20-year lifespan is not widely available. However, the turbines represented are all older models which are generally much smaller than their modern counterparts, with some being as small as 30 kW in the case of the WMEP study, and generally being less than 1 MW. As well as this, many of the turbines in the databases are older stall-regulated, fixed-speed designs. Because their configuration is quite different to modern turbines, the specific failure rates of individual assemblies is not too relevant, but the relative failure rate trends over age, size and geographic location are valuable.

All surveys showed that larger turbines were slightly less reliable than smaller

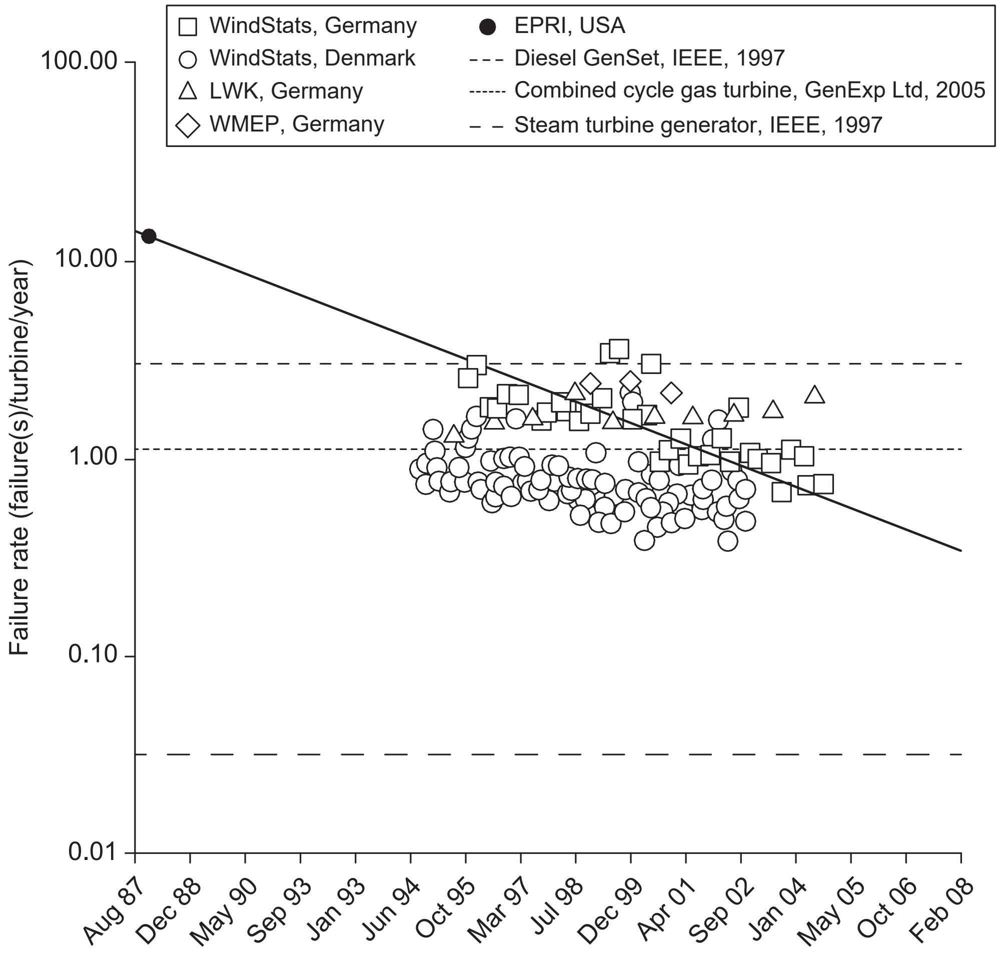


Figure 2.6: Failure rates by year of turbines in the WMEP, LWK, WindStats and Swedish databases reproduced by permission of the Institute of Engineering and Technology from (Peter Tavner 2012)

ones. The WMEP survey showed that the availability of turbines smaller than 500 kW was 98.5%, whereas the availability of turbines $\geq$ 1 MW was lower, at 98.4%. The study by Spinato et. al also showed that failure rates of turbines $\geq$ 1 MW of 2.4 to 3.4 failures per year, while smaller sub 500 kW turbines had failure rates of between 1.1 and 1.8 failures per year. This is to be expected, as larger turbines are inherently more complex machines.

All surveys also showed that turbines generally followed the "bathtub curve" as they aged. This is a theoretical function which shows a high failure rate early on in the operating life of a system, before it decreases and levels off for a period, and finally increases again as the system nears the end of its operating life-cycle. The WMEP survey tracked turbines in various years of operation and showed that although most assemblies loosely followed this trend, the electrical system clearly had more frequent failures as the turbines aged.

Interestingly, all of these populations showed overall increasing reliability with time. Figure 2.6 shows the failure rates by year of the turbines in each of the surveys. Despite larger turbines failing more frequently than smaller ones, turbines in general are becoming more reliable. This is likely down to more mature designs, and as Tavner notes, possibly due to newer wind farms being larger, and hence having a higher concentration of parts, personnel and tools close to the wind farm site.

2.6.2 ReliaWind

The EU-funded ReliaWind project aimed to further progress information on wind turbine reliabilities, resulting in a comprehensive study for more modern wind turbines (manufactured between 2006-2008 and at least 850 kW). Data was collected at a more granular level in terms of the reliabilities of specific sub-assemblies and components (Wilkinson et al. 2011). The project made a number of recommendations in terms of failure data collection, including mapping failures to a standardised taxonomy.

Due to being more modern wind turbines, there were more data sources available to the authors. A combination of SCADA operational and alarm data and some maintenance reports were used to build the database. Here, a failure is defined as an unplanned stoppage which lasts one hour or more and needed at least a manual restart to return to operation (Wilkinson et al. 2011). Different failure severities measuring 1-4 were attributed to each incident with 1 being a manual restart, and 4 being a major replacement.

The results of this study can be seen in figure 2.7. This shows the normalised failure rate for each assembly, broken down by the ReliaWind taxonomy. As can be seen, the most commonly failing assemblies were the pitch system, frequency converter, yaw system, generator and gearbox. The share of downtimes was broadly similar to this. The overall failure rate was roughly 2.67 failures/turbine/year. This is higher than more recent studies, but it should be noted that the definition of failure here included maintenance call-outs which resulted in a simple a manual restart.

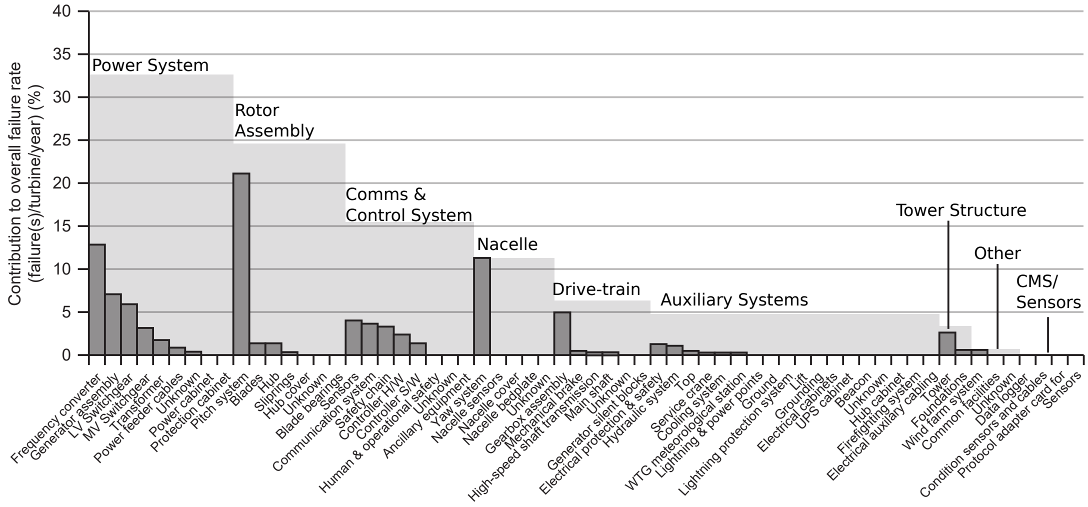


Figure 2.7: Failure rates of assemblies of turbines in the ReliaWind database, reproduced by permission of the Institute of Engineering and Technology from (Peter Tavner 2012). Data from (Wilkinson et al. 2011)

2.6.3 Reder et. al

Reder et. al (Reder et al. 2016) did a similar analysis to the ReliaWind study, but had a larger database and used a more modern taxonomy applicable to more modern turbines. Three years of data from turbines from a range of OEMs were analysed. The data here was taken from detailed work order databases by OEMs, so granular information was available. A failure was defined as any stop due to component failure that needed replacement and repair, i.e. automatic and manual restarts where no further action were taken are not included. The study comprised of over 4300 turbines, split according to their size and drive train configuration. There were roughly 2270 doubly-fed induction generator

(DFIG) turbines over $\geq$ 1 MW, and 215 direct drive (DD) turbines comprising between 300 kW and 3 MW.

Figure 2.8 shows the share of overall failure rate and downtime of each assembly across the DFIG turbines $\geq$ 1 MW. As can be seen, the most commonly failing components are the gearbox, controller and communications system, pitch system, cooling system, and power protection unit. The spread of downtimes here was quite different, with the gearbox, generator, blades and pitch system being by far the biggest contributors. This shows that the share of downtime for the gearbox, generator and blades is very high relative to the absolute number of failures, implying that when these systems fail, they tend to be more serious failures. In total, the failure rate for these turbines was .52 failures/turbine/year, with 112.67 hours of downtime/failure, or 44.51 hours of downtime/turbine/year. This is much lower than the ReliaWind study, but it should be noted that this may be because of the less stringent definition of failure that that study used. In general, findings were consistent with ReliaWind, with some small differences, for example the ReliaWind population had a much higher share of frequency converter and yaw system faults.

The spread of failures in the DD turbines was similar, but a higher share of the failures were attributable to the generator and electronic systems, with no failures in the gearbox due to the absence of this assembly. This is consistent with findings in (Carroll et al. 2015), as will be seen in section 2.6.4. Overall, there were .19 failures/turbine/year, with 34.98 hours of downtime/failure. This lower rate of failure may be because the direct drive turbines were as small as 300 kW, and, as was seen previously, smaller turbines have lower failure rates in general.

2.6.4 Carroll et. al - Drivetrains

Carroll et. al performed an extensive study of failure rates of the generator and converter for two turbine configurations. These are DFIG-configured turbines, with a partially rated converter (PRC), and permanent magnet generator (PMG)-configured DD turbines with a fully rated converter (FRC). The turbines represent 2,222 modern onshore turbines, built in the past five years. They are between 1.5 and 2.5 MW, with rotor diameters between 80 and 100 m.

The data comes from very detailed work order and inventory databases. A failure is defined as any unscheduled visit to a turbine where any consumables

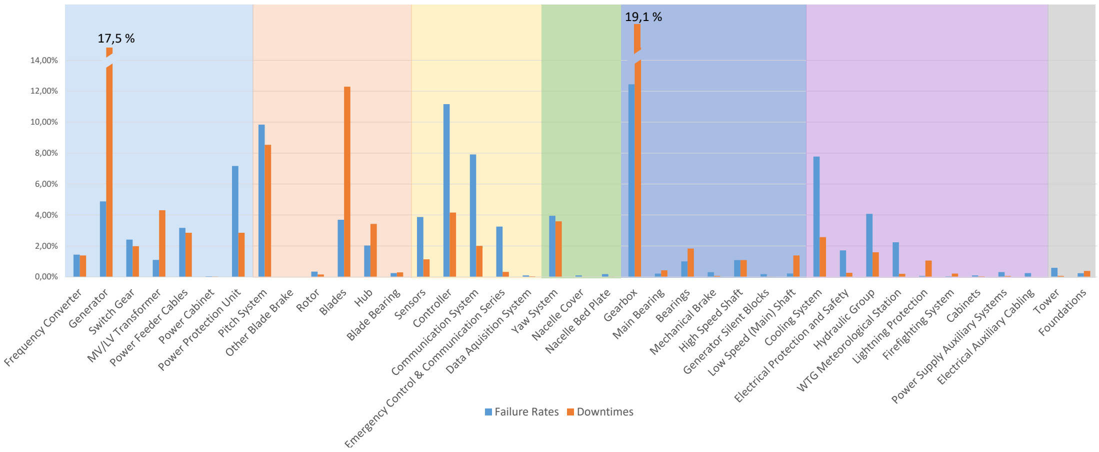


Figure 2.8: Failure rates of assemblies of geared onshore turbines ≥ 1 MW, reprinted from (Reder et al. 2016)

are used or parts replaced. Failures are also broken down into severity categories depending on the material cost of the repair or replacement (<€1000 for minor repair, €1000 – €10,000 for major repair and ≥ €10,000 for major replacement). Downtime for each of these failures was not available.

It was found that the DFIG failed slightly more than a PMG, but the massive failure rate of the FRC compared to the PRC meant that, overall, the failure rates of the PMG turbines' generator and converter (.669) was almost three times higher than for the DFIG turbines (.229). It is not known what the failure rates for other components were. Failures generally decreased over the first few years of operation of the turbines in the study, which is consistent with other studies.

Of particular note in this study is the cost breakdown, seen in figure 2.9. Major replacements of both configurations were very low - with just .014 and

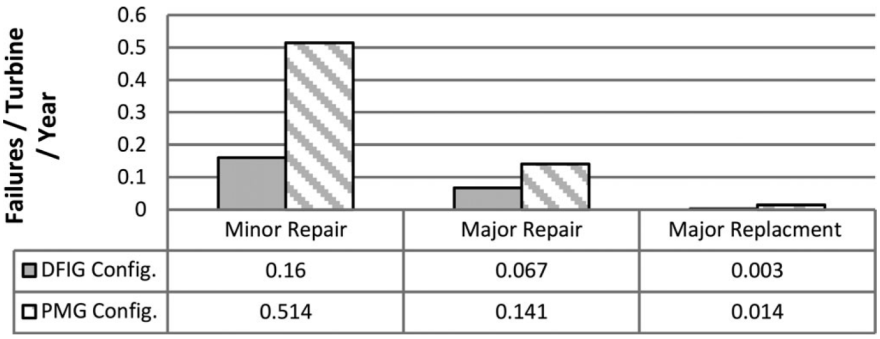


Figure 2.9: Combined generator and converter failure rate for DFIG and PMG-configured turbines, sorted according to cost, reprinted from (Carroll et al. 2015) © 2015 IEEE

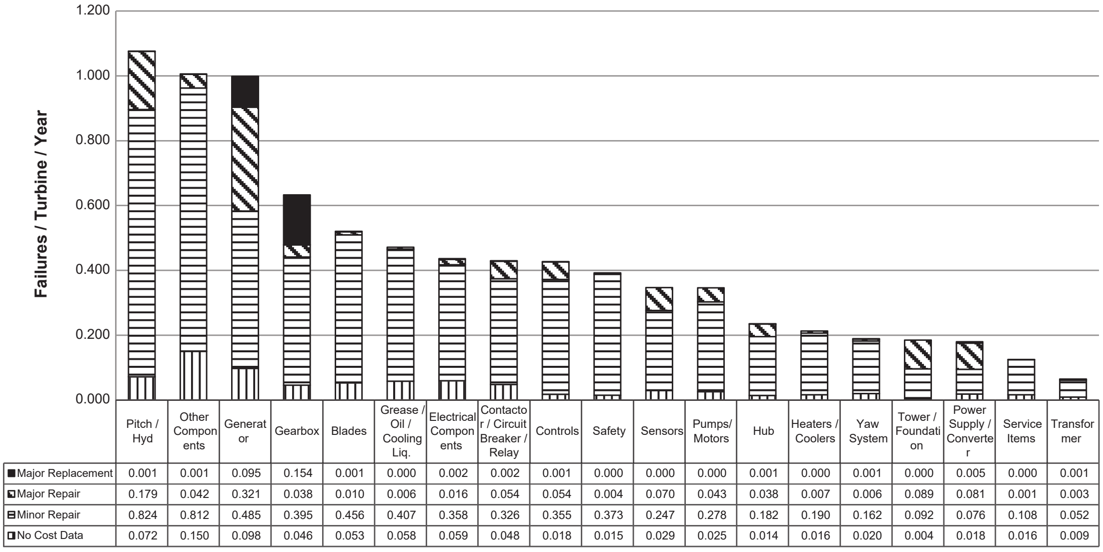
.003 major replacements/turbine/year for the PMG and DFIG configurations, respectively. This represented 2% and 1% of all faults for the respective configurations. Major repairs, meanwhile, accounted for 21% and 29% of failures, with the remaining 77% and 70% of failures being attributed to minor repairs.

2.6.5 Carroll et. al - Offshore

Carroll et. al also analysed the failure data of 350 modern offshore turbines from a number of wind farms throughout Europe in (Carroll et al. 2016). All turbines were between 3 and 10 years old and between 2 and 4 MW. Rotor diameters were between 80 and 120 m and 68% of the population was $\leq$ 5 years old. Similar to the analysis by Carroll et. al covered in section 2.6.4, the data came from a very detailed work order and maintenance inventory and costs database, so that highly granular information was available. The same definition of failure and different severity categories were also used, which depend on the material cost of the repair or replacement (i.e. do not count personnel or logistic costs, which can be very high for offshore turbines). Once again, down time was not available.

The results of this can be seen in figure 2.10, which shows the failure rate by assembly and also by severity category. As can be seen, the biggest contributor overall is the pitch and hydraulics system. This system also showed a marked increase in failure rate per year of operation for the turbines in the population. The next biggest contributor is "other components", which include ladders, hatches, lifts and door and nacelle seals. After this is the generator, gearbox and blades. Apart from the "other components", this is broadly consistent with other studies.

Where this study deviates from others however, is in the absolute failure rate. The failure rates of the turbines in the population ranged from 4 to 16 failures/turbine/year. This is much higher than the previously discussed studies which use a similar definition of a failure. The authors of this study conclude that some of this is down to the higher wind speeds seen offshore (consistent with findings in (Peter Tavner 2012)), but this cannot account for all of the increase. They posit that it may also be to do with access - onshore turbines can be accessed for routine PM much more easily and frequently. However, the authors also note that even accounting for these issues, there is still a marked difference in reliabilities. They conclude it may be to do with the fact that offshore turbines are a newer technology operating in harsher environments, and that there are factors unaccounted for which are driving higher failure rates offshore. This is also consistent with other studies, and is an issue which clearly needs to be addressed (Peter Tavner 2012, Maples et al. 2013).



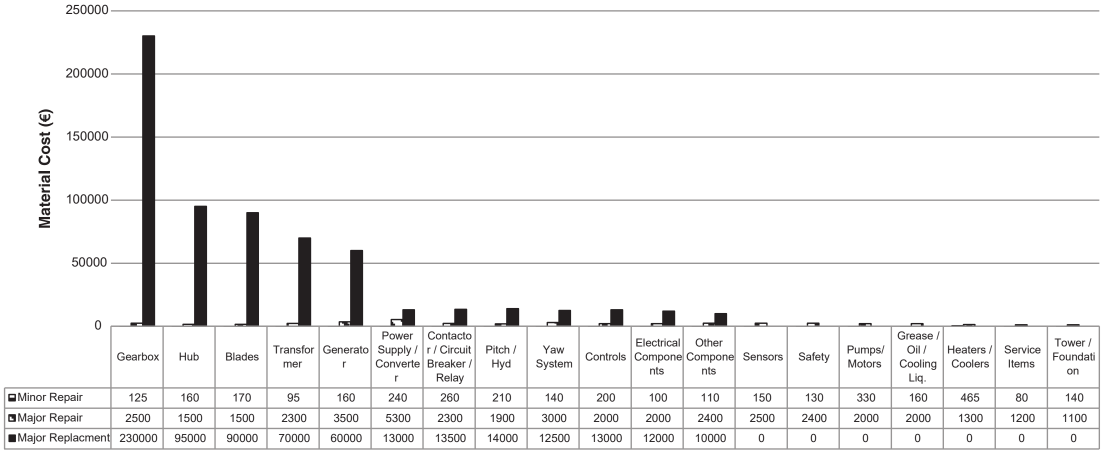
	Pitch / Hyd	Other Components	Generator	Gearbox	Blades	Grease / Oil / Cooling Liq.	Electrical Components	Contactor / Circuit Breaker / Relay	Controls	Safety	Sensors	Pumps/ Motors	Hub	Heaters / Coolers	Yaw System	Tower / Foundation	Power Supply / Converter	Service Items	Transformer
■ Major Replacement	0.001	0.001	0.095	0.154	0.001	0.000	0.002	0.002	0.001	0.000	0.000	0.000	0.001	0.000	0.001	0.000	0.005	0.000	0.001
▧ Major Repair	0.179	0.042	0.321	0.038	0.010	0.006	0.016	0.054	0.054	0.004	0.070	0.043	0.038	0.007	0.006	0.089	0.081	0.001	0.003
▤ Minor Repair	0.824	0.812	0.485	0.395	0.456	0.407	0.358	0.326	0.355	0.373	0.247	0.278	0.182	0.190	0.162	0.092	0.076	0.108	0.052
▥ No Cost Data	0.072	0.150	0.098	0.046	0.053	0.058	0.059	0.048	0.018	0.015	0.029	0.025	0.014	0.016	0.020	0.004	0.018	0.016	0.009

Figure 2.10: Failure rates of assemblies of offshore turbines, reprinted from (Carroll et al. 2016)

Once again, this study shows that, overall, the share of major repairs and replacements is quite low compared to minor repairs. However, as is consistent with the ReliaWind and Reder et. al studies, it seems that even though systems such as the gearbox and generator had less of a share of overall faults, their relative share of more serious faults was much higher. This is further reinforced in figures 2.11 and 2.12, which show the average repair cost and repair time for each category of failure in each of the assemblies. As can be seen, major replacements of the generator cost on average €230,000. This only includes the material cost; costs for offshore access and logistics are much higher than onshore, as will be seen in section 2.7. The MTTR (excluding LDT) for each category of failure was also investigated, and again major replacements in the hub, blades and gearbox showed that these were very significant maintenance operations taking a long time to complete.

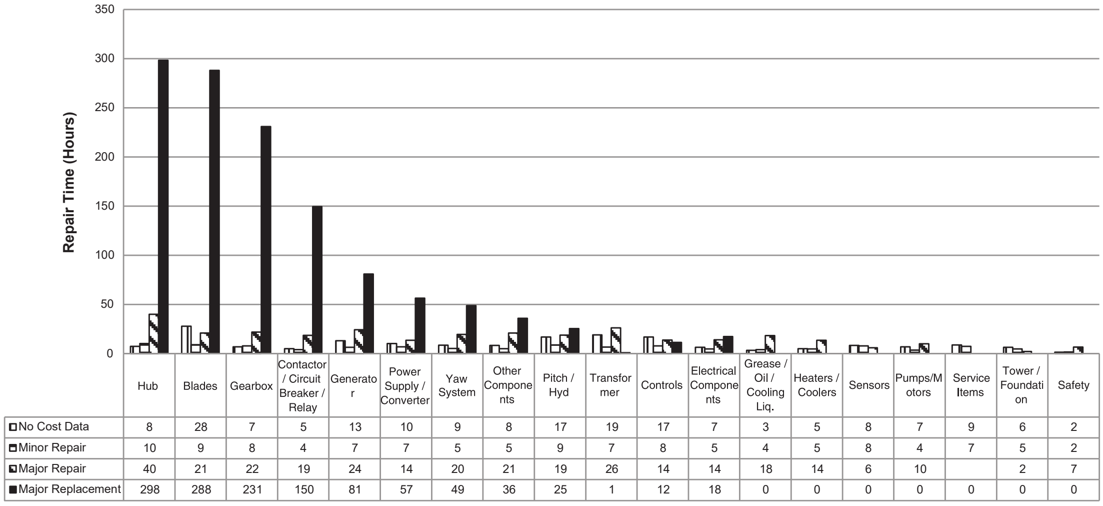
2.6.6 Reliability Studies Conclusions

All of the above studies show similar trends in the most commonly failing components (for both on- and offshore), and absolute average failure rates (for



	Gearbox	Hub	Blades	Transformer	Generator	Power Supply / Converter	Contactor / Circuit Breaker / Relay	Pitch / Hyd	Yaw System	Controls	Electrical Components	Other Components	Sensors	Safety	Pumps/ Motors	Grease / Oil / Cooling Liq.	Heaters / Coolers	Service Items	Tower / Foundation
⊟Minor Repair	125	160	170	95	160	240	260	210	140	200	100	110	150	130	330	160	465	80	140
◪Major Repair	2500	1500	1500	2300	3500	5300	2300	1900	3000	2000	2000	2400	2500	2400	2000	2000	1300	1200	1100
■Major Replacment	230000	95000	90000	70000	60000	13000	13500	14000	12500	13000	12000	10000	0	0	0	0	0	0	0

Figure 2.11: The average material cost to repair different categories of failure in each of the assemblies, reprinted from (Carroll et al. 2016)



	Hub	Blades	Gearbox	Contactor / Circuit Breaker / Relay	Generator	Power Supply / Converter	Yaw System	Other Components	Pitch / Hyd	Transformer	Controls	Electrical Components	Grease / Oil / Cooling Liq.	Heaters / Coolers	Sensors	Pumps/Motors	Service Items	Tower / Foundation	Safety
□No Cost Data	8	28	7	5	13	10	9	8	17	19	17	7	3	5	8	7	9	6	2
⊟Minor Repair	10	9	8	4	7	7	5	5	9	7	8	5	4	5	8	4	7	5	2
◪Major Repair	40	21	22	19	24	14	20	21	19	26	14	14	18	14	6	10		2	7
■Major Replacement	298	288	231	150	81	57	49	36	25	1	12	18	0	0	0	0	0	0	0

Figure 2.12: The average time to repair different categories of failure in each of the assemblies, reprinted from (Carroll et al. 2016)

onshore). What should be highlighted is that the downtime/repair costs associated with failures in major components such as the generator, gearbox or blades constitute a larger share of the downtime/repair costs relative to their respective failure rates. This implies that when these components fail, the failures are severe. This not only has implications for lost revenue, but also the materials cost for replacing these systems, as highlighted in (Carroll et al. 2015, 2016). This is further highlighted in the WMEP study, where it was found that 5% of fault events contributed to 75% of the downtime (Faulstich et al. 2011). Also of note is that the turbine pitch system contributes significantly to both down time and number of failures across all studies. This is a system which has not been a target of traditional CMSs, unlike the blades or drive-train.

Generally speaking, larger, more modern turbines see higher failure rates than older < 1 MW turbines due to their inherently more complex nature. However, the downtime per failure on newer turbines is much lower (Faulstich et al. 2008). As well as this, turbine failure rates seem to loosely follow the "bath tub curve", whereby failures are more frequent in early years of operation just after commissioning, before decreasing and levelling out for a number of years and then increasing again as components begin to age.

It is also important to distinguish between onshore and offshore failures. The reliability of modern offshore turbines was shown to be much lower than that of their onshore counterparts. Possible explanations for this are that offshore sites have higher wind speeds and harsher conditions and that these difficulties with access mean PM may not happen as frequently. However, being a relatively new technology, there may also be other, unaccounted for, factors which affect offshore components more than their onshore counterparts. As will be seen in section 2.7, the reduced reliability of components is compounded by access issues when performing corrective maintenance, which considerably reduces availability.

2.7 Wind Turbine Maintenance in Practice

During the warranty period, O&M is shared by the owner/operator and the OEM, with the OEM usually overseeing the main share of maintenance. After this period, the operator may choose to continue a service contract with the OEM, or may wish to impose their own asset management strategy, and use in-house or sub-contracted maintenance teams (Peter Tavner 2012). The specific maintenance strategy used will have a big impact on the long-term life and availability of the turbines; as stated previously, while reliability is dependent on component design, engineering and construction, turbine availability is dependent on the base reliability of the components coupled with a good maintenance strategy in order to minimise down time (Spinato et al. 2009).

This section will demonstrate current maintenance practices in the wind industry as they fit into some of the broad strategies outlined in section 2.1. It will be divided into two parts - the elements of the basic strategy employed by nearly all operators, and some more advanced CBM methods using dedicated sensors. These cover the most common types of maintenance carried out by operators and OEMs and are the prevailing methods found in industry and the literature. Although other more advanced methods may be in use commercially, due to their proprietary and necessarily confidential nature they cannot be effectively evaluated and will only be discussed in a limited fashion in this section. While this section discusses maintenance from a high level, strategic point of view, a robust review of the state-of-the-art techniques used in implementing some of these strategies (specifically, CBM as it relates to fault detection, diagnosis and prognosis based on SCADA data) will be given in section 2.8.

2.7.1 Basic Strategy

Almost universally, a preventive scheduled strategy is used to minimise unexpected failures, while corrective maintenance is performed when unexpected faults do occur. CM is widely practised in a rudimentary form throughout the wind industry; the turbine alarm system gives basic fault detection and diagnosis functionality, while most operators will also perform some high-level SCADA-based performance monitoring to detect any obviously under-performing turbines in a farm or fleet.

2.7.1.1 Time-based Preventive Maintenance

All turbines undergo scheduled PM carried out on a periodic basis in order to minimise unexpected failures, regardless of actual component condition (Link et al. 2011). The maintenance intervals are based on the historical reliability of various components as collected by OEMs, operators or industry groups. The turbine is usually brought off-line while diagnostics take place, and any necessary repairs are performed. This includes minor routine activities such as tightening and torquing of bolts or changing oil and filters. This intends to avoid any more serious failure modes which could cause catastrophic damage to the components. However, if major repairs or replacements are required, the turbine may be down for a number of days while the logistics of access and spare parts are organised, and the repairs performed (Walford 2006).

This preventive strategy is much cheaper than a run-to-failure model of corrective maintenance - Igba et. al found that using this type of preventive strategy over a 10 year period can be more than 3 times cheaper than a purely corrective strategy. They found PM carried out based on the historic reliability of the gearbox cost on average €110,000, while a corrective strategy over the same period cost in the region of €420,000 (Igba et al. 2014).

2.7.1.2 Corrective Maintenance

Assuming 100% site accessibility and efficient logistics (i.e. parts and crew availability), most onshore wind turbines achieve availabilities upwards of 98% using the preventive strategy outlined above. However, unexpected failures still do happen, and in these cases, corrective maintenance is employed. The most common type of unexpected shut down is a brief stop initiated by the control system after detecting some kind of potential fault. In these situations one of the actions outlined in section 2.5 must be carried out, and in the majority of cases result in a short shut-down followed by a remote reset without the need for an actual site visit or repair. As was seen in section 2.6, however, major failures, for example in the gearbox or generator, still do occur, and can be incredibly costly. Meanwhile, commonly occurring but less serious faults, e.g. problems with the pitch system, can contribute to extended periods of downtime.

The high availability of modern onshore plants is in part due to the very high reliability of onshore components - these components are showing reliabilities similar to components in other applications. Although this 98% may be further increased through increasing component reliability, the increased component cost may actually overall increase the LCOE (van Kuik et al. 2016). Meanwhile, as will be seen, the availability of offshore plants is significantly lower due to issues with access and logistics for faults which are comparatively simple to repair when they occur onshore. Hence, in order to increase availability both onshore and offshore, the focus should be put on improving the current maintenance strategies employed and eliminating unexpected failures, moving from a corrective to a purely proactive strategy.

2.7.1.3 CM by the Alarm System

The turbine alarm system can be thought of as a basic form of CMS, albeit one with only limited fault detection and diagnosis, and no prognosis, functionality. Since maintenance personnel do not normally have access to the turbine nacelle during operation, the alarm system gives a high-level indication of whether the turbine is operating normally or not. Some basic fault diagnosis functionality can also be given. For example, after the control system detects a potential fault and generates alarms for which an automatic or manual remote reset will not clear, the alarms can give an indication of the physical location or type of the potential fault before an on-site visit. In these cases, it can be useful to see if related alarms have been triggered over the previous weeks or months to determine the type of fault which may have occurred.

In a similar vein, a turbine may experience a number of shut-down/remote reset cycles over a period of days, weeks or months associated with a particular alarm or group of alarms. In these situations, although an on-site visit is not *required* to start up the turbine, the frequency of certain alarms may still be an indication there is an underlying issue. In these cases a maintenance crew may decide an on-site visit is necessary (Ribrant & Bertling 2007).

In both of these cases, particular types of alarms or sequences of alarms may have been previously identified by experienced technicians or maintenance teams through domain knowledge as being indicative of a particular type of fault. But because they are not always indicative of a fault having taken place, alarm systems do not provide full diagnostic capabilities, and the sheer volume of alarms generated can make them hard to fully utilise (Qiu et al. 2012).

Certain operators or manufacturers may also have their own in-house solutions for gaining more CM functionality from alarms and various information/warning messages to give indications of future faults. Some techniques present in the literature will be discussed in section 2.8. Some of the short comings of turbine alarm systems and proposed ways to improve these will be discussed in chapter 6.

2.7.1.4 SCADA-based Performance Monitoring

The final element that forms a part of the prevailing maintenance strategies is performance monitoring through the SCADA operational data, which can also be thought of as a limited form of CM. SCADA data is used by farm operators to get a high-level overview of plant performance at a site level. If specific turbines are noticeably under-performing in comparison to nearby turbines in a given period, action can be taken to investigate and root-cause any issues. This can include flagging with the OEM or maintenance contractor to investigate further, or drilling down into the data to see if there are any other anomalies. This performance monitoring usually includes looking at individual turbines' power curves on a monthly or weekly basis. In this way, if turbine performance starts to deteriorate, it can be indicative of deteriorating health, and hence shares some similarities to fault prognosis. However, performance deviation can be due to any number of issues, perhaps not directly related to turbine component health (e.g. incorrect noise or shadow settings), and hence in this manner does not approach the functionality of a dedicated CMS. Some of the techniques used will be discussed in section 2.8.

2.7.2 Dedicated CBM Strategy

Various investigations into turbine maintenance strategies, including RCM studies, have concluded that CBM is the optimal strategy for certain components and assemblies (Nilsson & Bertling 2007, Kandukuri et al. 2016, Andrawus et al. 2006). A secondary benefit to fault prognosis through CBM is that it helps operators give more accurate production forecasts, which is useful information to grid operators (Mc Garrigle & Leahy 2015). In many jurisdictions, electricity markets have moved or are planning to move to a model which requires non-dispatchable generators to give a forecast for their daily energy generation. For example, in Ireland, the new Integrated Single Electricity Market will have this requirement from 2018 (Eirgrid 2017).

There are a number of options open to operators which can enable a CBM strategy. These fit into two broad categories; "traditional" CMSs, which have seen wide success in other industries and are typically based on vibration sensors, and SCADA-based solutions, which have been developed in more recent times.

2.7.2.1 Traditional CMSs

As mentioned in section 2.2.3, more sophisticated on-line monitoring of turbines can be achieved through dedicated CMSs. These CMSs are usually supplied by third parties, outside of turbine OEMs, who specialise in these systems, and analysis and interpretation of the signals requires extensive domain expertise (Yang et al. 2014). Due to this, the CMSs are often operated by the third parties themselves. They can either be pre-fitted to turbines in partnership with turbine OEMs, or retrofitted afterwards at extra cost. While these systems have seen success in other industries such as marine propulsion and conventional generation, they are not yet ubiquitous in the wind industry (Walford 2006, Link et al. 2011).

There are a number of reasons for this. Wind turbine components in general are quite reliable, and with costs upwards of €14,000 per turbine, installing a CMS can take up a significant portion of a turbine's annual maintenance budget; the revenue stream from an individual turbine may be an order of magnitude lower than where a CMS is applied in other industries (Yang et al. 2014, Sheng & Veers 2011). Because specialist knowledge is needed to interpret the data generated, operators may need to rely on a service contract with the OEM or third party to provide this analysis, which can incur further costs (Peter Tavner 2012). Furthermore, due to the stochastic nature of the wind, and the fact that turbines operate at much lower and variable speeds under a variety of conditions, the signals are harder to interpret and have not seen the same successes as in other industries (Yang et al. 2013). A previous NREL round-robin study provided sixteen industry CMS providers with measured CM data from a gearbox which had experienced a number of faults. Each of the industry partners performed their own analysis, with results across the board indicating a number of missed faults and false alarms, as seen in figure 2.13 (Sheng 2012). Nonetheless, CMSs are still effective in many cases, and the techniques are continuously being refined and have significant presence in the literature (Tchakoua et al. 2014, García Márquez et al. 2012, Lu et al. 2009). However, this thesis will focus on the application of SCADA-based CM due to some inherent advantages of such an approach, detailed next.

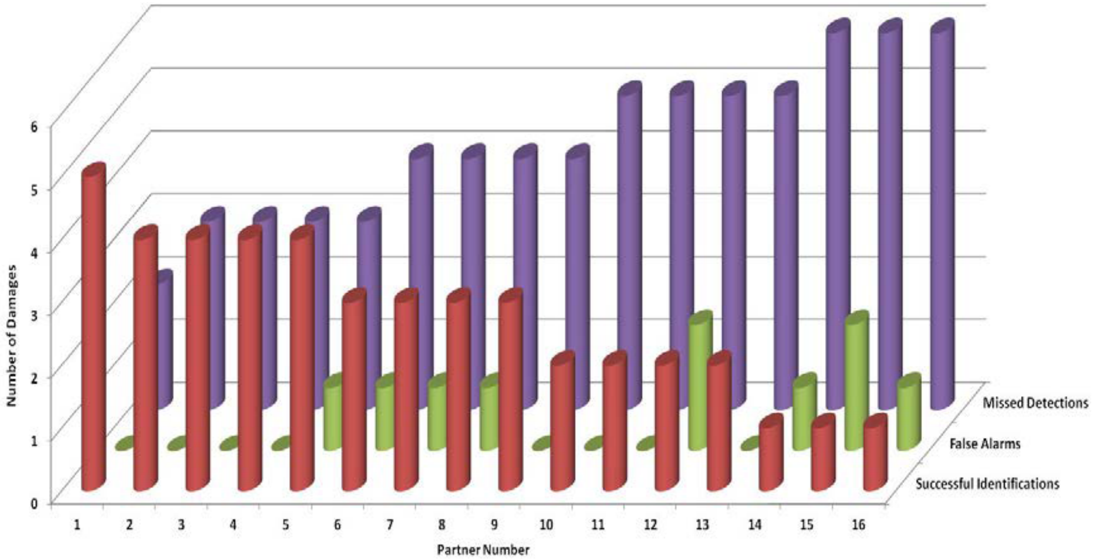


Figure 2.13: Results of the NREL round robin gearbox CM study, reprinted from (Sheng 2012)

2.7.2.2 SCADA-based CMS

Advanced monitoring of SCADA or alarms data has seen some success in industry. These techniques largely rely on using artificial intelligence methods to detect, diagnose or predict faults. Even without a full RUL estimate, the ability to detect that a specific type of fault may be imminent has significant utility for maintenance teams. The fact that these techniques leverage existing SCADA data for CM, rather than installing a number of dedicated extra sensors, has a number of advantages. First, SCADA data already covers all major components in the turbine - the control system monitors sensors from all major assemblies, including assemblies which are not usually targeted by traditional CMSs, such as the pitch system. Hence a SCADA-based CMS potentially has wider coverage than traditional CMSs. Related to this, the fact that SCADA-based CM does not require the installation of additional sensors significantly reduces the up-front capital cost of such a system. Finally, SCADA data is usually stored at a 5- or 10-minute resolution, compared to the kHz+ frequency of traditional CMS data. Hence, a SCADA-based CMS can significantly reduce data bandwidth and transmission costs. While some of these more advanced methods are now becoming available to operators (Greenbyte AB 2017), the state-of-the-art as seen in literature will be discussed in section 2.8.

2.7.3 A Note on Offshore Maintenance

While an onshore corrective maintenance operation, from the fault being flagged to the repair being finished and the turbine operating again may be measured in hours, for an equivalent fault offshore this can stretch to days (Richardson 2010). This is mainly due to issues with access; offshore repair operations are highly sensitive to weather and sea conditions which greatly increases the LDT (Feng et al. 2010). This also means that the downtime associated with less severe faults is much higher than onshore; a simple on-site inspection and manual reset can involve significant logistical planning compared to onshore, increasing the share of the 5% of downtime being caused by 75% of faults in the WMEP reliability study (Faulstich et al. 2008).

Transport costs for access to offshore turbines are also much higher than their onshore counterparts. For example, transport for a maintenance crew can range from €85/hr on relatively small vessels which are highly dependent on calm seas, to over €400/hr for larger vessels and €1200/hr for some types of helicopters (Peter Tavner 2012). This is anywhere from 1.5 to 25 times the cost of an onshore maintenance activity (Richardson 2010). Furthermore, excluding material costs, the complexity of major replacements of components such as the gearbox offshore involves specialised vessels and cranes which further drive up the cost (van Kuik et al. 2016).

In all, these issues have significant implications for availability of offshore turbines. These availabilities, in the region of 84%-93%, are significantly lower than the 98% seen onshore (Maples et al. 2013, Stehly et al. 2016). As seen in section 2.6, the reliability of offshore components is much lower than onshore, but it is unclear how much of this is due to reduced accessibility for PM, and how much is due to engineering design. What is clear, however, is that offshore availability needs to be raised to have a competitive LCOE. Hence, corrective maintenance must be minimised, and a clear CBM strategy will be key in achieving this (Nilsson & Bertling 2007). An NREL study concluded that, assuming 50% of failures are detected within a sufficient PF-interval, the use of a vibration-based CMS could increase offshore availability by 1.2% (Maples et al. 2013). This echoes conclusions drawn by the EU FP7-funded LEANWIND project, which stated that the use of vibration-based CMS shows a clear cost benefit for offshore wind (LEANWIND 2017). Both of these studies noted the limitations and costs of current vibration-based CMSs and that increased benefits could be seen as technology matures.

2.8 SCADA-based CM and Fault Prediction Techniques and Methods

As discussed in section 2.7, traditional CMSs have not seen as much success in the wind industry compared to other industries, due to significant up front costs and difficulties with the stochastic nature of wind generation. Turbine SCADA systems, on the other hand, record a wide range of operational data which has conventionally been used for high-level performance monitoring. This has generated much research interest in recent years, as leveraging this data can provide a very cheap alternative to a traditional CMS. Because the data is typically at a low resolution, the infrastructure needed for transmitting and processing is also much lower. Hence, if any level of CM can be provided by SCADA data, whether it fully captures the full functionality of a traditional CMS or not, there is a clear cost benefit to its use. This section will provide an overview of the current research landscape in using wind turbine SCADA data to achieve the varying levels of CM functionality as described in section 2.1.4.1. In most of the techniques discussed, 10-minute SCADA data is used, though some techniques based on alarm system data are also presented.

2.8.1 Trending

A straightforward way of determining wind turbine health is to track the trends of various parameters over time to see if any obvious change can be visually observed. The thinking behind this is that any obvious change in a trend can be indicative of an obvious change in the internal physical state of some turbine component(s), and hence warrants further investigation to diagnose the source of the change. In this way, basic trending is comparable to fault detection, but not diagnosis or prognosis.

In its most basic form, trends can be visually observed by turbine operators, who use domain knowledge and experience to detect performance deviations, or diagnose faults. For example, diagnosing obvious changes in the shape of the power curve over different periods of time can be indicative of control system issues (Lindahl & Harman 2012). As mentioned previously, this is widely practised in industry. However, subtle changes in the shape can be hard to detect, so a more analytical approach can yield better results.

One way of doing this is visually comparing values across turbines at the same site. In (Wilkinson et al. 2014), the authors showed that comparing the on-line temperature values of specific components across turbines can show deviations from the norm. A time series of normalised temperature of a drivetrain component of a test turbine was compared to the average of four control turbines. It was shown that the temperature of the test turbine began to noticeably deviate from the control turbines component temperature in the weeks leading up to a fault in the drivetrain. However, it was noted that this method did not perform as well as methods which used physical or data-driven normal behaviour models of the temperature.

Another approach is to use first principles equations to determine a physical representation of the system under normal conditions. Feng et al. derive equation 2.1 for representing the temperature rise across the gearbox compared to the

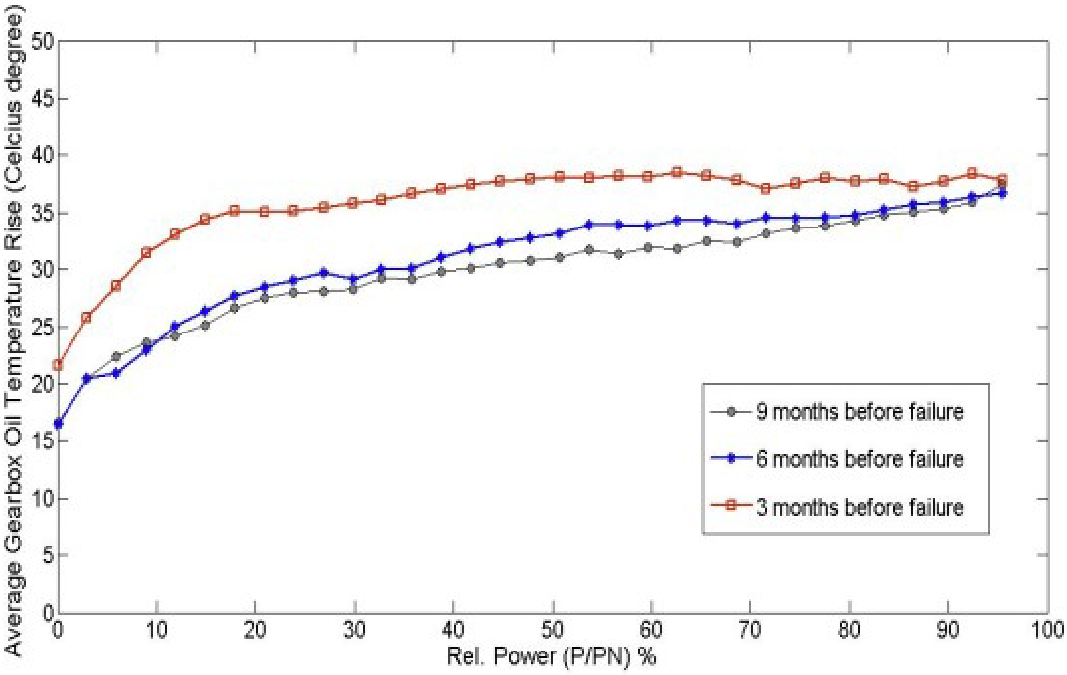


Figure 2.14: Trend of gearbox oil temperature against normalised power output for different months in advance of gearbox failure, reprinted from (Feng et al. 2011)

base nacelle temperature for different power outputs of the turbine (Feng et al. 2011).

$$\Delta T = P_{out}\frac{1}{k_{gear}}(\frac{1}{\eta_{gear}} - 1) \tag{2.1}$$

Where ΔT is the temperature rise, P_{out} is the power output of the turbine, η_{gear} is the efficiency of the gear transmission, and k_{gear} is a constant. Hence, because the gear transmission efficiency should be constant, the temperature difference should not change for a given power output under normal operating conditions. Should the gear condition deteriorate, the efficiency should decrease, leading to a proportional decrease in the gearbox oil temperature rise. The results of this for measurements of 3, 6 and 9 months in advance of a known gearbox failure are shown in figure 2.14. The temperatures are binned in 40 kW increments, and the power output normalised. As can be seen, the results clearly show a trend leading up to failure.

Being able to quantify how different a trend behaves from the norm allows operators to view the significance of performance deviation. In (Yang et al. 2013), a metric, c, is developed for various trends to show the correlation of historic to present data, which is then used to quantify performance deviation from this historic norm. Curves of various trends, e.g. power output vs. generator speed or torque vs. rotational speed were plotted for normal behaviour and different stages of a fault, as measured by a test rig. An example of this can be seen in figure 2.15, which also shows the c metric increasing as a winding fault develops.

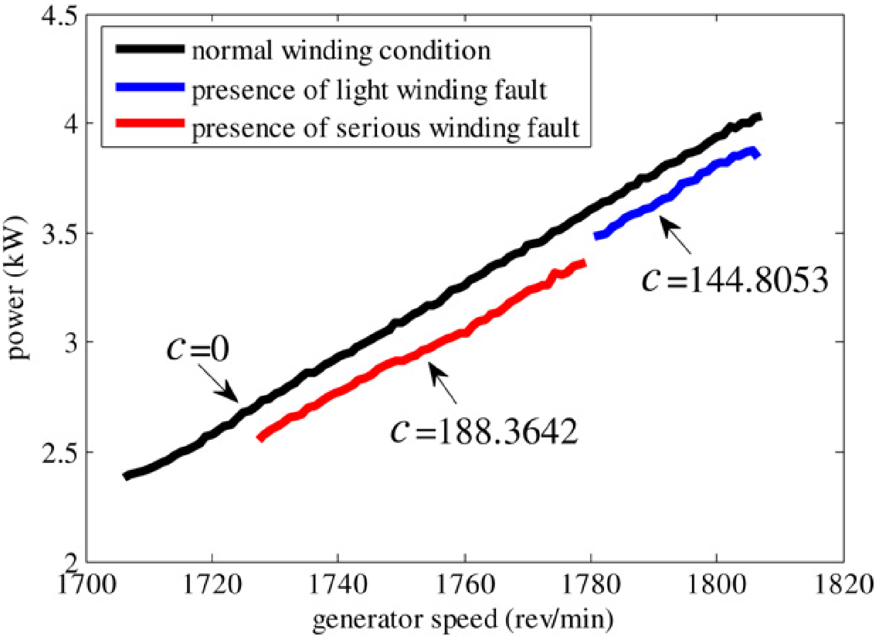


Figure 2.15: Power output vs. generator speed for various modes of operation, showing a correlation metric, c, of the curves to nominal values (lower values indicate higher correlation), reprinted from (Yang et al. 2013)

As shown, observing general trends in turbine data can show indications of faults before they occur. These trends can be easy to implement, however the disadvantage of such approaches is that they rely on human visual interpretation and domain knowledge. Further, specific trends related to different types of faults must be tracked. This leads to a high workload for the operator, with little scope for automatic alarm generation or automation.

2.8.2 Normal Behaviour Modelling

Normal behaviour modelling (NBM) relies on building a model of the system under consideration during normal operation. The output being modelled can be the turbine power output, component temperatures, or some linear or non-linear combination of a number of parameters representing the system. Inputs to these models can be any number of SCADA parameters, including autoregressive (AR) components, in which past values of the output are used as current inputs. Models can be data-driven or physical models. Data-driven models use historical values of inputs and outputs to model the system, whereas physical models use governing equations or simulations derived from first principles. One of the challenges when using data-driven NBMs is filtering historical data so that only "fault-free" data is used. Different works use different approaches to address this.

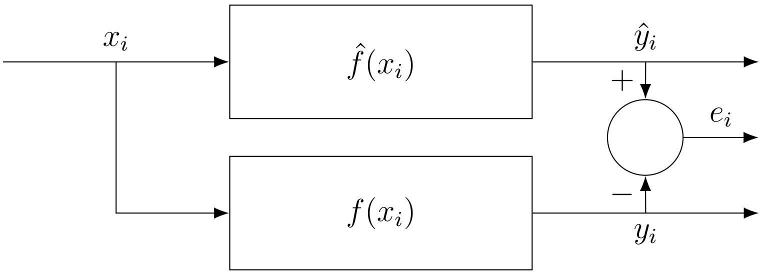


Figure 2.16: NBM overview, showing the on-line inputs, x_i, the actual system and output, $f(x_i)$ and y_i, the modelled system and output, $\hat{f}(x_i)$ and $\hat{y}_i$, and the residual or error signal, e_i

Fault detection is performed by comparing real on-line values of the system to expected modelled values, and quantifying the difference between the two. A common way of doing this can be seen in figure 2.16, showing the on-line inputs for sample i, x_i, the actual system and output, $f(x_i)$ and y_i, the modelled system and output, $\hat{f}(x_i)$ and $\hat{y}_i$, and the residual or error signal, e_i.

Note that, for data-driven models, the historical data is split into a "training" and "testing" set. The training set is used to build the model, while the test set is used to evaluate its performance, and get an approximate measure of real-world performance. If the same data is used to both train and test the model, the model's performance will be exaggerated in a problem known as "over-fitting" (James et al. 2013).

Any significant residual between the model and actual output can be indicative of a developing or materialised fault. A simplified example of this is shown in figure 2.17. This shows a control chart, whereby the residual for each sample is plotted, and a certain threshold is used to highlight anomalous values. A "sliding window" is used to aggregate samples, whereby if 3 samples in a window of length 10 exceed the specified alarm limit for the residual, an alarm is generated. The first window, w_1, shows such an alarm. In the case of w_2, only a single sample exceeds the limit so an alarm is not generated. The generated residual can either be a simple difference between y_i and $\hat{y}_i$, or, if the output is multi-dimensional, some distance metric between the two points in space can be used.

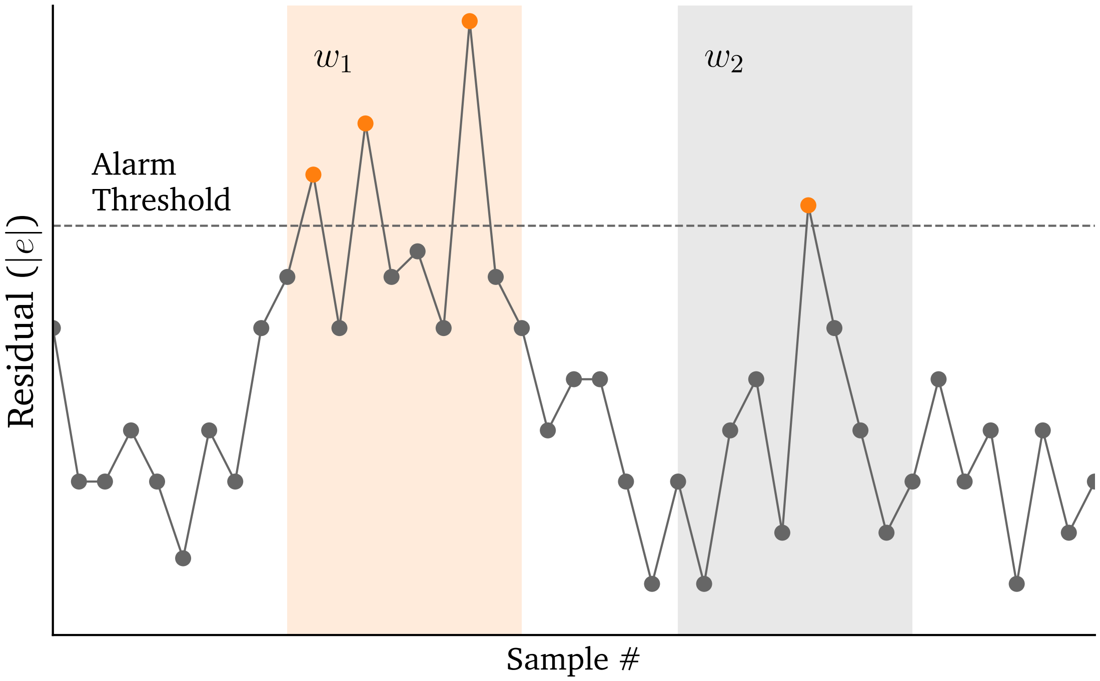


Figure 2.17: Control chart showing the alarm limits of the residual $|e_i|$ being exceeded. If three or more samples in a sliding window exceed the limit, an alarm is triggered, as in w_1. An alarm is not triggered in w_2 does not trigger the alarm due to only a single sample exceeding the threshold.

2.8.2.1 Data-Driven Performance Monitoring

A widely practised NBM strategy is to model a turbine performance metric such as the power curve. The rationale behind this is that some physical degradation leading up to a major component fault is expected, and this degradation will lead to a deviation in performance. In (Butler et al. 2013), the authors first filter out any non-normal values by only looking at the linear portion of the power curve. They then filter out other values by only including SCADA samples within certain blade pitch angle limits, and finally filter out outlier wind speeds and frozen anemometer values. The remaining values represented 60-70% of all historical data used to build the model. Data-driven Gaussian process models were then used to statistically model the power output based on historical air density and wind speeds. A separate set of data for each turbine was used to simulate on-line values and build the residual, and the cumulative sum of the error was plotted over a period of 14 months. It was found that in a fault-free turbine, the cumulative sum oscillated between roughly -10 and 5 scaled units, while in a turbine with a main bearing fault, the residual built to -25, before increasing again after the turbine returned to full service. No alarm limits were derived.

In (Skrimpas et al. 2015), the authors use kernel methods to build similarity indices to compare the power curves of different operating turbines. The similarity index was summed using a sliding window metric to build a cumulative residual. It was found that the residual showed anomalies corresponding to periods of ice build-up on turbine blades, control system issues, and periods of de-rating.

In (Park et al. 2014), a novel algorithm is developed for modelling the power curve. The average power output at various different wind speed "bins" is found. A provisional power curve is then built by interpolating between these points. Next, optimal bounds are developed by shifting this curve up and down by varying degrees. All points outside these bounds are filtered out, and the process repeats itself until a satisfactory model representing nominal operation is found. Control charts are created by giving the upper and lower bounds specified "alarm" limits. When a number of consecutive on-line samples lie outside the limits, an alarm is generated. This method showed an indication of faulty operation, but did not diagnose a specific fault.

In (Cambron et al. 2016), the authors use only data where the turbine was available and producing power, and further filter out outliers using methods inspired by the work in (Park et al. 2014). The IEC 61400-12-1 (International Electrotechnical Commission (IEC) 2013) method of binning is used to build a reference power curve, though the authors note that there are better methods to model power curves, such as those mentioned in a comprehensive review in (Lydia et al. 2014)). Exponentially weighted moving average (EWMA) and generally weighted moving average (GWMA) control charts were then used to detect any under-performance. It was found that both EWMA and GWMA control charts were able to detect a 1% per year under-performance. A case study on a test turbine showed that a historical 5-year 3.4% under-performance caused by blade erosion could have been detected in advance in time for maintenance to be carried out.

Kusiak et. al compare a number of different machine-learning methods of modelling the power curve based on wind speed in (Kusiak et al. 2009). Outliers

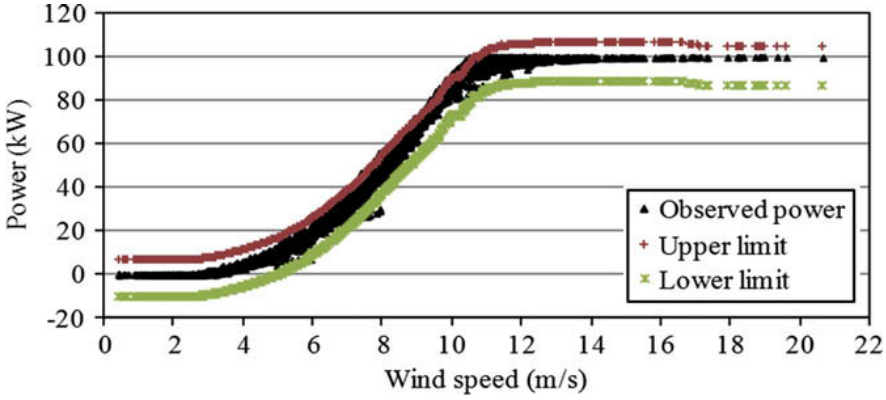


Figure 2.18: Power curve showing actual values, and control chart limits based on modelled values, reprinted from (Kusiak et al. 2009)

are removed from the training data, before training data is used to model the power output using a least squares method, maximum likelihood estimation, and a number of non-parametric approaches, including multi-layer perceptron and random forests. Upper and lower limits of acceptable power output for each wind speed were used to build control charts, as shown in figure 2.18. In this way, any values which resided outside the control charts were highlighted as anomalous. This study did not contain any information on possible reasons for the anomalous values.

In (Schlechtingen et al. 2013), the authors compared a number of machine learning techniques, including adaptive neuro-fuzzy interference system (ANFIS), cluster centre fuzzy logic (CCFL), k-nearest neighbour (k-nn), ANNs and the M5P decision tree algorithm. The power output was modelled based on ambient temperature and wind direction as well as wind speed in all cases. It was found that the error signal increased in advance of faults, but the authors noted that identifying small deviations in the 10-minute averaged SCADA values still proved a challenge.

Work by Lapira et. al built baseline models of power output using wind speed as well as various component temperatures, rotor speed and pitch angles as inputs. When compared to live values, it was found that an ANN based model showed a spike in the residual weeks before a major downtime event (Lapira et al. 2012).

In (Uluyol et al. 2011), the authors fitted the power curve using polynomials, and found that looking at deviations in the mean, baseline and kurtosis of baseline compared to on-line values of the power curve shows that there are noticeable performance improvements after corrective maintenance actions have taken place.

In (Kusiak & Verma 2013), the authors expanded the idea of using NBMs based on the power curve to include other performance curves such as the rotor curve (wind speed vs. rotor speed) and blade pitch curve (wind speed vs. blade pitch angle). They use k-means clustering to split each curve into a number of different "zones, and use the Mahalanobis distance (MD) to filter out outliers from each zone. The MD can be thought of as a measure of the distance between two points in space which happen to be correlated. This leaves filtered reference curves which indicate normal behaviour. The skewness and kurtosis of the reference curves are then transformed to a single metric using Hotelling's T^2 chart, and this is where the residual is built. Results showed that deviations in performance were detectable and corresponded to periods of power curtailment or forced shut-downs due to cable untwisting, etc.

2.8.2.2 Data-Driven NBM of Component Temperatures

By expanding the space of input parameters to the model, and modelling parameters specific to particular components, such as temperatures, more granular fault detection is possible. In (Garlick et al. 2009), the authors used a linear AR model based on the generator winding temperature, power output and wind speed to predict the generator bearing temperature. Normal periods of operation were found through the use of fault logs. Differences between modelled and on-line values of bearing temperature were found in some cases to correspond to entries in the fault logs.

The authors in (Cross & Ma 2014) used both a dynamic AR exogenous (DARX) and ANN model to build a non-linear representation of turbine operation. They used a simulated turbine to show that a grid fault could be predicted from a residual of predicted vs. simulated current. Real SCADA data was also used to model gearbox bearing temperature, using both power and wind speed as inputs. The residual between the on-line vs. modelled temperatures was found to increase during fault events.

In (Butler et al. 2012), the authors used a set of radial basis functions trained using sparse Bayesian learning to model the main bearing temperature of a set of wind turbines operating under fault-free conditions. They improved the performance of the residual by applying a low-pass filter, and use this to give an RUL estimate by applying particle filters. The fact that Gaussian models were used enables a probability density function (PDF) to be built, giving a confidence score for the RUL estimate. They found that the RUL estimate becomes accurate as the actual main bearing failures approach, as seen in figure 2.19. The authors note, however, that the limited amount of failure examples in the study means the technique must be evaluated further in order to be robust.

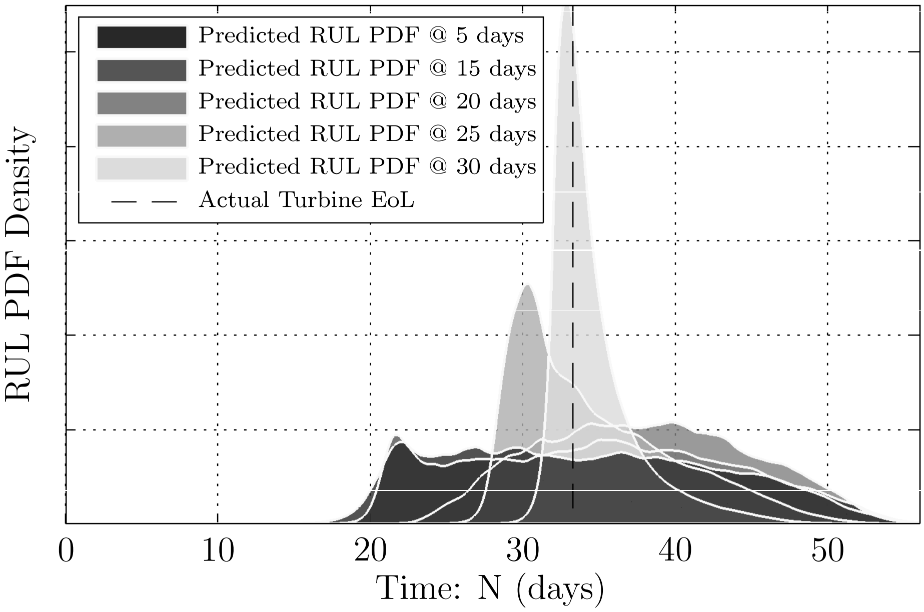


Figure 2.19: Evolution of the RUL PDF at different days in advance of the end of life of the main bearing, reprinted from (Butler 2012) © 2012 IEEE

Many approaches use ANNs to build the NBMs. Kusiak and Verma in (Kusiak & Verma 2012) attempt to find over-temperature faults in the main bearing. They use a number of data-mining algorithms to find optimal model parameters, before training a number of different ANNs to model the temperature. The models were successfully able to detect anomalous temperatures up to 90 minutes before certain over-temperature events which occurred over a week of data. However, because the models were tested on a single week of data with a high number of fault instances, it is not known how well it would perform over a longer period of time (i.e. whether there would be more false-alarms in periods where there were zero fault instances).

Tautz-Weinert and Watson in (Tautz-Weinert & Watson 2017*b*) complement an ANN NBM with a physics-of-failure-based approach. The ANN is trained using 3 months of data from an operating wind farm, and a sliding window metric is used to generate alarms in the residual. The specific labels of the temperature signals being modelled were not available to the authors, so they use a correlation analysis to see which ones are most likely to correspond to fault instances.

Alarms are generated if 3 out of 10 samples in the window exceed certain limits. A number of wind turbine performance parameters, including counts of high wind speed, generator rotational speed and wind turbulence intensity are analysed to see if there are any discernible patterns in these parameters leading up to faults. Results show that the NBM throws alarms leading up to gearbox and bearing failures, but also shows a number of false alarms. The physics-of-failure approach did not show any distinct patterns in the distribution of performance parameters leading up to fault events, leading the authors to conclude that there were a wide range of failure modes in each of the recorded faults.

In (Bangalore & Tjernberg 2015), the authors built various ANN models of the temperatures for five gearbox bearings. The training data was selected by selecting a subset of samples from the historical data which match the distribution of the full set, as per previous work by the authors in (Bangalore & Tjernberg 2013). Inputs included the output power of the turbine, gearbox oil temperature, nacelle temperature, generator speed, an AR component, as well as temperatures of other gearbox bearings. The average MD over three days between the on-line and modelled temperature was used as a residual, with the threshold defined by training results. A gearbox failure was identified one week before a traditional CMS detected the fault, highlighting the effectiveness of such an approach.

In (Schlechtingen & Ferreira Santos 2011), the authors showed that using an ANN to model various component temperatures showed evidence of an increasing residual leading up to the failures of five major components. No alarm limits were set, so this increasing residual was visually observed.

2.8.2.3 Other Data-Driven NBM Techniques

Other data-driven techniques build a representation of the system that does not model a specific output. One approach is to use deep autoencoders (DAs) to model normal behaviour. These are an unsupervised ANN whose objective is to learn a representation of the input parameters x_i, and reconstruct them as $\hat{x}_i$. The residual is then the difference between a sample x_i and the reconstructed sample $\hat{x}_i$. In (Wang et al. 2016), the authors built a DA model with 25 different SCADA parameters, including wind speed, power output, generator and rotor speed, various drive-train temperatures, voltages and currents, in order to detect blade breakages. Once built, residual was generated based on the square of the Euclidean distance between modelled and test values. EWMA control charts were then used to highlight anomalies, and results showed that blade breakages were detected between 2 to 8 hours in advance. Interestingly, in this study a separate blind validation was performed where the models were trained with a full set of data (i.e. fault data was not filtered out). Even so, the charts managed to pick up the two cases of blade breakages present in the data, with no false alarms (though in this case the advance notice was a maximum of 2 hours). It should be noted that it was not clear in this study whether the data used to train the models was separate from the data used to evaluate them, which can lead to models which have been over-fit, and an inflated performance compared to real-world deployment.

Other approaches use a technique known as self-organising maps (SOMs). This is an ANN method used to cluster together similar points in space. Clusters are built with normal operating data, with neurons acting as cluster centres. A distance metric is used to calculate similarity between a new sample and its closest neuron. When a sample is dissimilar to all of the existing clusters, it can be indicative of a fault. Du et. al used SOMs to build a baseline model of turbine performance, and showed that some deviations were visible in advance of faults occurring, while other deviations were due to faulty sensor values (Du et al. 2016). Wilkinson et. al used a similar approach in (Wilkinson et al. 2014) to find abnormal modes of operation, but here again, no diagnosis of the location or type of fault was possible. The authors of both of these papers note that SOMs are limited to detecting abnormal turbine behaviour, but not diagnosing the source of this abnormality. This is similar to power curve-based methods.

2.8.2.4 NBM Based on Physical Modelling

Because data-driven NBMs rely on accurate input data in order to build the models, a physics-based representation of the system can be a better approach when this data is hard to get or incomplete. In this case, the model $\hat{f}(x_i)$ is built from a physics-based understanding of the system.

In (Godwin & Matthews 2014), the authors use the temperature model in equation 2.1, developed in (Feng et al. 2011), to build NBMs of gearbox oil temperature rise. Robust MD (RMD), a slight modification to MD, was then used to build a residual, and control charts used to highlight anomalies at a warning and alarm level. It was found that although there were a number of individual instances of false alarms, the RMD value consistently stayed above the alarm level prior to a serious gearbox fault. Because this metric was quite accurate, data was labelled as fault-free, inspection suggested, and potential damage (according to no warning, warning, or alarm status), and the RIPPER propositional rule-learning algorithm was trained on the data to classify future points with human-readable rules. The results showed a high precision and recall across all three classes.

In (Borchersen & Kinnaert 2016), the authors developed mathematical models of the generator heating and cooling system. Although the specifics of the system were not known, the model was parameterised using a Kalman Filter. Anomaly detection was performed by building residuals across 3 cooling coils, using a cumulated sum of differences. Results showed that 16 out of 18 cooling coil faults over 3 years of SCADA data were detected, with a single false alarm.

In (Wilkinson et al. 2014), the authors built a hybrid physical and data-driven model of drive train component temperatures. A multiple linear and polynomial regression equation was built from a generic energy balance representing a non-specific component. The polynomial degree in the equation was selected by knowledge of the physical system, and the coefficients were solved using training data from periods of normal operation. The model highlighted an increasing residual in various gearbox bearing temperatures up to 9 months before gearbox failures. The physical NBM was compared to an SOM-based NBM, as well as simple trending of temperatures, and was found to have the best performance overall.

2.8.2.5 NBM Conclusions

NBM can be an effective way of achieving CM in turbines. Unlike trending, it allows a quantifiable way of evaluating the change in behaviour or performance of a turbine over time. Some of the power curve-based approaches can be straightforward ways of building turbine performance metrics without the need for extensive data filtering and analysis. This means they can be easier to implement than more specific or complex NBMs. The major disadvantages of using such an approach is that deviations in the power curve can be fairly small, and so issues may only be detectable after a cumulative residual has built up over a period of weeks or months. Furthermore, fault diagnosis is not possible when using a simple performance metric such as the power curve; although the fact that a turbine may be under-performing can be detected, the underlying reason for it is hidden.

Other approaches use a wider selection of parameters to model specific attributes such as component temperatures. Some of these approaches have successfully demonstrated anomalous behaviour, for example higher than expected component temperatures, weeks or months in advance of a failure, solely based on 10-minute SCADA data. However, in a number of cases no specific alarm limits were set, and so rely on visual diagnosis. In other cases, only a small subset of the data was used to test the algorithms, so that the true distribution of faulty to fault-free data was not preserved in the testing phase.

The components analysed using temperature-based NBM typically fail infrequently, and so "normal" training data used to build the models is relatively easy to obtain. This is in contrast to power curve data, where the power output can fluctuate due to a number of causes, e.g. periods of grid-mandated de-rating. The temperatures of major components can still be modelled as a function of generator RPM, power output, etc. during these events, so that this behaviour is safely captured in temperature models. As will be seen, getting access to "clean" fault-free data is challenging, so this provides a major benefit. However, temperature-based models can only be applied to specific components, and separate models must be trained for each one.

Techniques which build a representation of the system without modelling a specific output, such as DAs or SOMs, showed some success, though these techniques only show that the turbine is performing abnormally, as opposed to diagnosing the source of the fault. Physics-based models, meanwhile, avoid needing to get access to a large volume of historical data, but are only applicable to specific components which can be effectively modelled in this way.

Overall, NBM is an effective way of providing CM functionality to wind turbines, and is analogous in many ways to the techniques that traditional vibration-based CMSs use. However, in order to achieve prognostic functionality relating to specific components, models must be developed for each individual component being monitored. Furthermore, NBM of some frequently failing assemblies, such as the pitch system, is not present in the literature. This may be because there are no obvious SCADA parameters which can be modelled in this way.

2.8.3 Alarm System

As described in section 2.7.1.3, the turbine alarm system already provides some CM capabilities in the form of limited fault diagnostics. As will be discussed in chapter 6, however, turbine alarm systems generally generate alarms at a much higher rate than in other industries. The techniques described here attempt to address some of these issues to enhance the diagnostic capabilities of turbine alarm systems.

In (Qiu et al. 2012), the authors acknowledge the issue of high alarm volume and found that the quantity of alarms raised is generally too high for operators to effectively manage. This is even higher during fault periods, when alarms occur in large and complex "showers", making failure detection, location and diagnosis difficult. By performing probability and time-based analyses, alarms which regularly appear together or trigger one another can be identified, and from this it is possible to identify particular failure modes related to alarm sequences. Hence, more advanced fault diagnosis is possible.

In (Chen et al. 2011), the authors acknowledge the problems associated with the volume of alarms being generated by wind farms. They aimed to reduce this by using pattern recognition techniques based on artificial neural networks (ANNs) to identify alarm patterns which occurred during or leading up to particular fault events. The authors found that by processing any alarm occurrences through the ANN model, faults could be flagged without the need for an operator to try and analyse the alarms themselves, thereby reducing the number of alarms that operators must work with. However, they noted that further work is needed to improve the accuracy of this method.

As discussed in section 2.4, mapping turbine alarms to a standardised taxonomy can help build a better picture of how faults propagate through various systems and understand the root cause of turbine stoppages (Reder et al. 2016, Peter Tavner 2012). In (Gonzalez et al. 2016), the authors show that by mapping turbine alarms to this taxonomy and performing a similar probability analysis to that of (Qiu et al. 2012), alarms which are directly related to faults can be identified and the propagation of a fault from root cause to failure mode can be manually established.

2.8.4 Classification-based approaches

2.8.4.1 Background to Classification

Classification is a machine learning technique that maps an input sample, x_i, to an estimated qualitative output class $\hat{y}_i$. Similar to NBM, this mapping is done through a function approximation, $\hat{f}(x_i)$. $\hat{f}$ is *learned* from a historical set of n pairs of training observations, $\{x_i, y_i\}_{i=1}^{n}$, where the vector $x_i \in \mathbb{R}^p$ contains the p features associated with the sample, i.e. $x_i = (x_{i1}, ..., x_{ip})^T$. In the field of wind energy, these features could represent SCADA parameters such as wind speed, power output, etc. Meanwhile, y_i is the actual class associated with x_i, e.g. "no fault" or "potential generator fault". The notation x_j refers to the j-th feature of all samples, i.e. $\mathrm{x}_j = (x_{1j}, ..., x_{nj})$ (James et al. 2013). The matrices X and Y are used to represent the full set of samples and features and their associated classes. Note that, similar to data-driven NBM techniques, data is split into a train and test set.

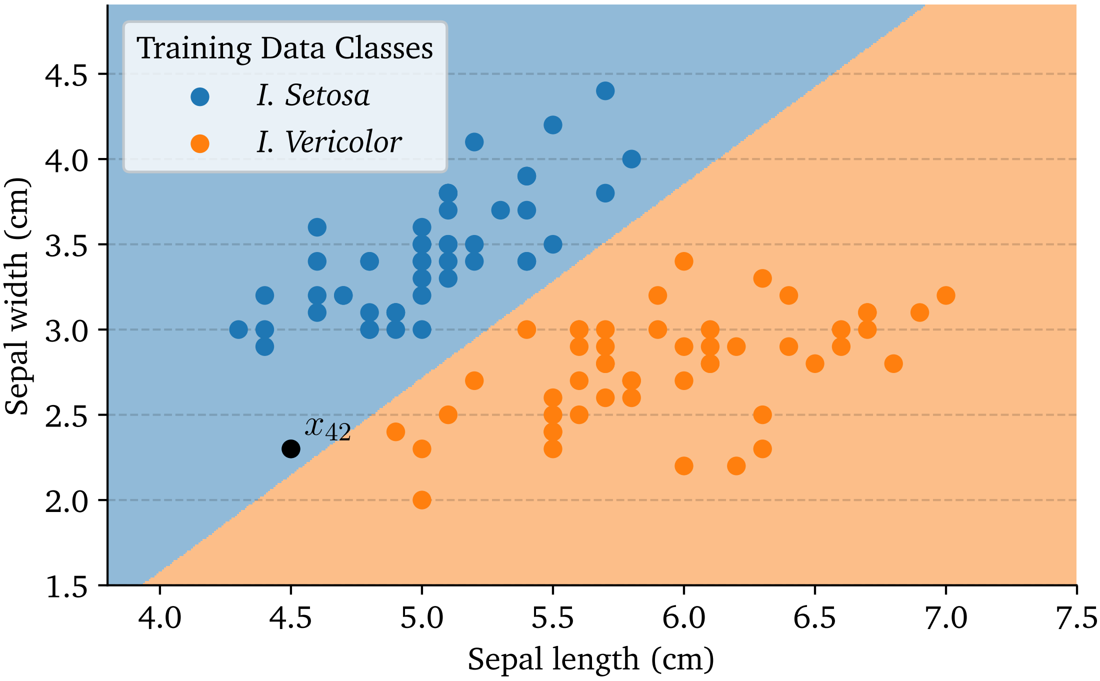


Figure 2.20: Plot of samples in the Iris dataset, showing the decision boundary between the two classes

A well-known classification problem, first proposed by Sir R. A. Fisher in 1936, is demonstrated here to give a high-level overview of classification. An attempt is made to try and classify different types of *Iris* flowers, based on the length and width of their *sepals*, (FISHER 1936). Here, the $p = 2$ features, x_1 and x_2, represent *sepal length* and *sepal width*. The possible values of y are *I. Setosa* and *I. Vericolor*. A linear classification technique known as *logistic regression* is applied to the training data to build a decision boundary between the two classes, shown in figure 2.20. A test point, $x_{42} = (4.5, 2.3)^T$, is shown to be above the decision boundary, and so $\hat{f}(x_{42}) = \hat{y}_{42} =$ *I. Setosa*. Note that, as will be seen, this can be generalised to any number of dimensions, so for $p > 3$, such a decision boundary can not be effectively visualised.

2.8.4.2 Success in Other Domains

Classification techniques have seen success in many other domains, which form similar problems to fault detection in wind turbines. Work done in (Esteva et al. 2017) attempted to identify different types of skin cancer based on photos of skin lesions. The resulting models were able to effectively distinguish between two different types of skin cancer in one case, and distinguish a particularly deadly form of melanoma from a benign nevi in another. Performance was on par with that of clinical dermatologists, potentially providing for low-cost access to diagnostic care.

In (Phua et al. 2004), the authors showed that applying a classification-based approach to insurance claims can be an effective way of detecting fraudulent claims. The model used inputs such as the month or day of week the claim was made, the make or model of car, or whether a police report was filed, among others. Results showed that insurance companies could save up to 30% per year by using such a model.

Fault diagnosis in other industries is also common. In (Abbasion et al. 2007), the authors use a classification technique known as a support vector machine (SVM) to identify faults in rolling element bearings. The features were based on vibration signatures, and results showed the model could distinguish between normal state and 7 different types of faults with 100% accuracy.

In (Susto et al. 2015), the authors attempt to build a PdM system for wafer manufacture. They label m samples in advance of a fault sample as being part of the fault class, for many different values of m. A cost, J is assigned to the percentage of missed failures and number of process cycles that could have been run had maintenance not been carried out, and train a number of classifiers (in this case also SVMs) to minimise J for different values of M. They devise a way of simulating the cost of PM, and perform a number of Monte Carlo simulations to compare the two approaches. They found that the classification-based PdM approach performs better than a number of PM approaches.

In all of these cases, the samples in the positive class(es) (i.e. those representing faults or unhealthy samples) were massively outnumbered by the negative class (i.e. fault-free or healthy). Hence, these applications, particularly in the cases of fault detection and prediction where time-series data is used, draw many parallels to the wind energy domain.

2.8.4.3 Classification Metrics

There are a number of different scoring metrics important to classification. Some of these which are especially relevant to CM of WTs are given below, where tp is the number of true positives, i.e., correctly predicted fault samples i test set, fp is false positives, fn is false negatives, i.e., fault samples incorrectly labelled as no-fault, or as a different fault-class, and tn is true negatives.

A high number of false positives, i.e. fault-free data incorrectly classed as faulty or fault-imminent, can lead to unnecessary checks or corrections carried out on the turbine. The avoidance of this is captured with the precision score (where a higher score represents a lower false positive rate):

$$Precision = tp/(tp + fp) \tag{2.2}$$

A high number of false negatives, on the other hand, can lead to failure of the component with no detection having taken place (Saxena et al. 2008). This is captured by the recall score (also known as sensitivity), where a higher number indicates a low ratio of false negatives:

$$Recall = tp/(tp + fn) \tag{2.3}$$

The F1-Score, which is the harmonic mean of precision and recall, is a good way of capturing both of these measures in a single metric:

$$F1 = 2tp/(2tp + fp + fn) \tag{2.4}$$

Specificity measures the proportion of true negatives that are correctly identified as such:

$$Specificity = tn/fp + tn \tag{2.5}$$

Confusion matrices can also be used to give a visual overview of performance and show absolute numbers. An example confusion matrix is shown in figure 2.21. It is immediately obvious from this image that the classifier has performed well, showing the power of this type of visual tool.

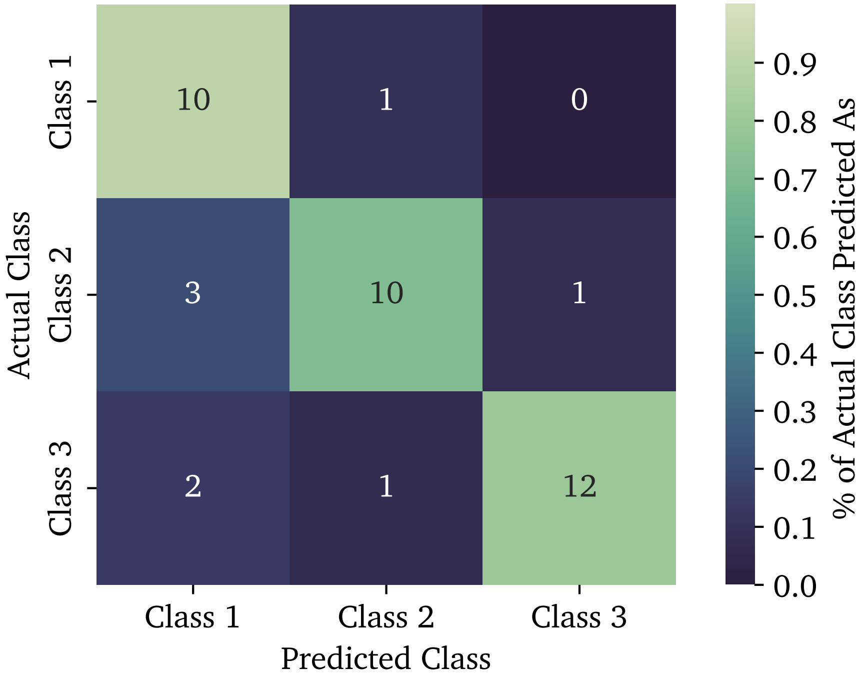


Figure 2.21: A sample confusion matrix, showing the absolute numbers and ratio of correctly classified samples in each class

Overall accuracy (i.e. $tp + tn/total$) is not a useful metric in the domain of CM, as there is a massive imbalance in the classes; any fault-related classes will be massively outnumbered by the fault-free class in the test set. For example, if 4990 samples were correctly labelled as fault-free, and the only 20 fault samples were all also incorrectly labelled as such, the overall accuracy of the classifier would still stand at 99.6%.

2.8.4.4 Classification-Based Approaches in the Literature

The concept described in figure 2.20 can be expanded so that y can represent a particular type of fault or pre-fault state in a turbine, while x can represent a number of SCADA parameters. The advantage to such an approach over using NBMs or other methods is that a number of different turbine states can be learned at once, i.e. separate models do not need to be trained and tuned for individual components, with the caveat that the quality of data must be high.

In (Kusiak & Li 2011), the authors use wind turbine alarm system data to label periods of faulty operation on the turbine. A number of models, comprising of ANNs, boosting trees, support vector machines and standard classification regression trees, were built to evaluate their performance in predicting and diagnosing faults. Overall, it was found that the most successful algorithm for specific fault prediction was the boosting tree algorithm. It was found that prediction of a specific fault, a diverter malfunction, was possible at 68% accuracy, 73% recall and 66% specificity 30 minutes in advance of the fault occurring. Unfortunately, when this was extended out to one hour in advance, accuracy and recall fell to 49% and 24%, respectively. The full results can be seen in table 2.3. Unfortunately, precision was not provided as a metric, so insight into the number of false alarms was not given. It should also be noted that the test set used for fault prediction in this paper did not represent the distribution of the underlying labelled SCADA data. Instead, the fault-free class was undersampled so that there were just over twice as many "fault-free" instances as there were of a specific fault. This is in contrast to the true distribution where there could be up to 100 times more fault-free than fault instances.

Table 2.3: Results for prediction of a specific fault at various 10-minute time steps in advance of a fault from (Kusiak & Li 2011)

Time Stamp	Accuracy (%)	Recall (%)	Specificity (%)
t	69.81	86.67	63.16
t - 1	64.15	66.67	63.16
t - 3	67.92	73.33	65.79
t - 6	67.92	73.33	65.79
t - 9	66.04	33.33	78.95
t - 12	49.06	24.53	34.21

A similar approach was used in (Kusiak & Verma 2011). Here, turbine alarms related to blade pitch faults were identified through data mining algorithms. Times when these alarms were present were then used to label the data as "faulty" or "normal". Genetic programming was used to classify the data, and performance on a test set showed that blade pitch faults could be predicted up to 10 minutes in advance with 68% accuracy, 71% recall and 67% specificity.

This work did indeed use a full set of data for the test set, though the data here was at a resolution of 1s rather than the more common 10 minutes. Once again the precision score was not provided, so the number of false positives could not be effectively evaluated.

Chen, Matthews and Tavner showed that pitch faults could be predicted with high precision and accuracy with a prognostic horizon of up to 21 days using an Adaptive Neuro-Fuzzy Inference System trained on SCADA data (Chen et al. 2013). However, the data in this study only included timestamps within a window of 7, 14, or 21 days before corrective maintenance actions occurred. This meant that although there were definite prognostic indicators of pitch faults in the data, the system was not tested on a full set of data, where time between corrective maintenance actions could be much larger than 21 days. This mean that the ratio of fault-free to faulty data is much higher in a real world scenario, so the study did not reflect real-world performance.

Godwin et. al used the RIPPER decision tree algorithm (Cohen 1995) to derive human readable rules for classification of pitch faults (Godwin & Matthews 2013). The authors found that pitch faults were detectable in a window up to 48 hours in advance of a fault occurring, with precision and recall scores of 0.8 and 0.75, respectively. However, these scores were obtained on a dataset which once again undersampled the fault-free class. When tested on a full set of data, precision scores dropped to .17, but there was still an overall reduction in the number of false alarms compared to the turbine alarm system by 52%.

2.8.4.5 Conclusions and Comparisons to Other Approaches

It is clear from the literature that for classification-based approaches, although there is clear evidence that faults can be predicted in some capacity, evaluation on a full test set representing the true distribution of live data has not been performed in all cases. Furthermore, it is important that the false positive rate be represented somewhere in performance metrics, which was also not included in all cases. Hence, the performance of such approaches compared to techniques such as NBM has not been as robustly demonstrated in the literature.

As mentioned previously, a classification-based approach to CM has some inherent advantages over NBM and trending. In some NBMs, the condition of the system as a whole is measured, for example in performance monitoring, using DAs, or using SOMs. Hence, fault diagnosis is not possible. Some leading signatures of faults are detectable, but in these cases, the only information available to operators are indications that the condition of a component is deteriorating or has deteriorated, but which component, or which fault mode is likely to occur, is not. At this level, simple performance-based NBMs may be more suitable than a classification-based approach, depending on the data available.

By building NBMs of specific parameters like component temperatures, diagnosis, and even some prognosis, is possible. However, classification has the inherent advantage of being able to monitor or detect a number of different faults at once, as performed in (Abbasion et al. 2007) and (Kusiak & Li 2011). This means that a number of different types of faults can be trained in a single model; classification based on thousands or more classes in domains such as image recognition has been demonstrated in the literature (Gupta et al. 2014). Related to this point, classification can also perform more granular fault diagnosis than NBM on a single component. For example, in (Kusiak & Verma 2011), where a number of different faults are diagnosed in the turbine pitch system. Finally, classification can be applied to components for which NBM is not appropriate, as in (Godwin & Matthews 2013) where faults in the pitch system were predicted.

2.9 Conclusion

This chapter has served as a background and problem formulation, detailing the need for more advanced maintenance strategies than those currently employed on wind turbines, as well as discussing the relative merits and shortcomings of some solutions. A background to maintenance theory was given, describing some common strategies and the reasons they are employed. An overview of the data available to farm operators was also given, followed by a detailed review of wind turbine component reliabilities and overall availability from both an onshore and offshore perspective. This review showed the most commonly failing wind turbine components across a number of different reliability studies, as well as the downtimes associated with these failures. This demonstrated the need to avoid major failures, and outlined the compounding issues that affect offshore turbines, underlining the significantly reduced availability resulting from access issues offshore. The review also demonstrated some of the challenges that researchers face when attempting to leverage disparate sources of turbine data to identify fault periods on turbines, which will be further expanded upon in chapter 4.

Current turbine maintenance strategies were also discussed, demonstrating that there is scope for improving availabilities through improving these strategies, particularly in the offshore context. Using the turbine SCADA system data was highlighted as a cost-effective way of achieving some CM functionality, without the need to retrofit expensive dedicated CMSs. A number of approaches showing varying levels of functionality were reviewed. Classification was highlighted as a technique that has seen broad success in other industries, with more limited success in wind energy despite some of its advantages over other methods. Also highlighted were some approaches which seek to enhance the diagnostic capabilities of the turbine alarm system. Chapters 3, 4 and 5 will attempt to expand on some of the classification methods described, as well as proposing a methodology for effective labelling of fault data, while chapter 6 proposes a system to glean more useful diagnostic information from turbine alarms.

Chapter 3

A Case Study on Classification Techniques

3.1 Introduction

It is clear from the literature review in section 2.8 that the SCADA system can be successfully used for all three levels of CM: fault detection, diagnostics and prognostics. Some indication of complete failure of main components were, in some cases, detected weeks in advance through the use of NBMs, for example in (Butler et al. 2012) and (Schlechtingen & Ferreira Santos 2011). However, also mentioned were comparative benefits of using classification-based approaches over NBMs, namely; being able to train a single model to perform CM on a number of different components at once; to give a more granular level of fault diagnosis; and, being able to detect faults in systems for which NBM may not be appropriate. This last point covers components where less serious, but more frequent faults occur. These types of faults, such as electrical or pitch system faults, were shown to have a significant share of turbine down time, and hence, availability, in section 2.6. This was shown to be particularly true in the offshore context.

A number of classification-based approaches present in the literature were discussed. However, in many cases, the studies did not use an appropriate metric for quantifying the number of false-positives in the data, and/or did not evaluate the performance on a held-out test set (i.e. a test set which did not under-sample the fault-free class, and represented the true distribution of data as seen in the real world).

In this chapter, a case study is presented in which classification techniques are applied to wind turbine SCADA data in order to perform CM at the detection, diagnosis and prognosis levels. Also investigated are a number of techniques which attempt to address issues with the massive imbalance seen in the distribution of samples in each class, due to the fault-free data being considerably dominant.

The fault detection level aims to differentiate between fault/non-fault state, while diagnosis aims to determine which type of fault has occurred. Although the turbine alarm system already provides this aspect of CM functionality, having an additional system which corroborates alarm information can give additional confidence to operators that the diagnosis is correct. This is particularly important in the offshore context, as noted in section 2.7.3. The fault prognosis level attempts to give an alert that a specific type of fault is likely to occur within a certain time window. Although not a conventional RUL estimate, it can be thought of as a window of possible RUL. This is advanced functionality that is not currently available through the alarm system.

3.2 Support Vector Machines

Although a comparison of a number of classification methods in (Kusiak & Li 2011) found a boosting tree algorithm to be more successful than other methods, including Support Vector Machines (SVMs), for CM applications in wind turbines, the authors did not go into detail on the specifics of the models used. However, SVMs are a widely used and successful tool for classification. They are very well suited to this specific problem, where the relationship between a high number of parameters (e.g., the many different parameters collected by a SCADA system) can be complex and non-linear (Cortes & Vapnik 1995, Boser et al. 1992). They have been used in other industries for condition monitoring and fault diagnosis with great success. A review by (Widodo & Yang 2007) showed that SVMs have been successfully used to diagnose and predict mechanical faults in HVAC machines, pumps, bearings, induction motors and other machinery. CM using SVMs has also found success in the refrigeration, semiconductor production and chemical and process industries (Laouti et al. 2011).

The basic premise behind the SVM is that an optimal decision boundary is made between two opposing classes, based on labelled training data. If a training set

S is given by $S = \{x_i, y_i\}_{i=1}^{n}$, where $x_i \in \mathbb{R}^p$ represents the n input samples and p features, and $y_i \in \{-1, 1\}$ represents the two opposing classes, then the goal of an SVM is to find an optimal hyperplane separating the two classes, subject to:

$$\begin{aligned} w^T x_i + b &\geq 1 \qquad \text{for } y_i = 1 \\ w^T x_i + b &\leq 1 \qquad \text{for } y_i = -1 \end{aligned} \tag{3.1}$$

where $w \in \mathbb{R}^n$ is a weight vector and b is a scalar representing the bias. If eq. 3.1 is true for all training data, then the training set will be a linearly separable case (Hastie et al. 2009). An example of such a case can be found in figure 2.20. However, should there be a case where no perfect hyperplane can be found to separate the data, or to avoid the problem of over-fitting, a number of points can be allowed to be misclassified by introducing *slack variables*, ϵ_i, and a scalar *cost penalty*, C. In this case, the optimal hyperplane can then be found by solving the following optimisation problem:

$$\text{Minimize} \quad \frac{1}{2} w^T w + C \sum_{i=1}^{n} \epsilon_i, \tag{3.2}$$

$$\text{Subject to} \quad y_i(w^T x_i + b) \geq 1 - \epsilon_i, \quad i = 1, ..., n \tag{3.3}$$

$$\epsilon_i \geq 0, \qquad i = 1, ..., n \tag{3.4}$$

The Lagrange dual of this problem becomes:

$$\text{Minimise } L(w, b, \alpha) = \frac{1}{2} w^T w + C \sum_{i=1}^{n} \epsilon_i - \sum_{i=1}^{n} \alpha_i y_i (w^T x_i + b) + \sum_{i=1}^{n} \alpha_i, \tag{3.5}$$

where the $(\alpha_1, ..., \alpha_n)$ arise from constraint 3.3. The task now becomes one of finding the saddle point of this function with the equations:

$$\frac{\partial L}{\partial w} = 0, \quad \frac{\partial L}{\partial b} = 0 \tag{3.6}$$

which then become

$$w = \sum_{i=1}^{n} \alpha_i y_i x_i, \quad \sum_{i=1}^{n} \alpha_i y_i = 0 \tag{3.7}$$

By substituting 3.7 into 3.5, the following optimisation problem is formed:

$$\begin{aligned} \text{Maximise} \quad & \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j x_i^T x_j, & (3.8) \\ \text{Subject to} \quad & \sum_{i=1}^{n} \alpha_i y_i = 0, \qquad i = 1, ..., n & (3.9) \\ & 0 \leq \alpha_i \leq C, \qquad i = 1, ..., n & (3.10) \end{aligned}$$

Once the optimisation problem has been solved, the coefficients α_i can be found, which then leads to the linear decision function for a new sample x_m, $x_m \in \mathbb{R}^p$:

$$f(x_m) = \text{sign}(\sum_{i=1}^{n} \alpha_i y_i (x_m^T x_i) + b), \tag{3.11}$$

A powerful aspect of the SVM algorithm is that it can solve non-linear cases by mapping the input vector x_m to very high dimensional space where it can then be solved linearly. Such a function takes the form $\Phi(x_m) = (\phi_1(x_m), ..., \phi_l(x_m))$ for l-dimensional space. While working in high-dimensional space allows complex decision boundaries to be formed, it can also lead to expensive computational overhead. An approach known as the *kernel trick* allows this overhead to be avoided through the use of kernel functions. These functions, of the form $K(x_i, x_j) = (\Phi^T(x_i)\Phi(x_j))$, compute the *inner products* of samples in the feature space, instead of explicitly computing Φ to get the coordinates of samples in that space. This means learning in the high dimensional feature space can be achieved more efficiently, allowing SVMs to build complex non-linear decision boundaries in a practical way. The decision function eq. 3.11 then becomes:

$$g(x_m) = \text{sign}(\sum_{i=1}^{n} \alpha_i y_i K(x_m, x_i) + b) \tag{3.12}$$

The kernel function $K(x_m, x_i)$ can take on a number of forms for creating non-linear decision functions. These are any functions which satisfy Mercer's theorem (Cortes & Vapnik 1995). Some of the most common kernels include the polynomial kernel:

$$K(x_m, x_i) = (1 + x_m^T x_i)^P, \tag{3.13}$$

where P represents the polynomial degree; and the Gaussian kernel (also known as the radial basis function):

$$K(x_m, x_i) = \exp(-\frac{||x_m - x_i||^2}{2\sigma^2}), \tag{3.14}$$

where σ represents the variance of the data.

The SVMs in this work used a "one-against-one" approach for multi-class classification, i.e. classification of more than two classes. For K different classes, $K(K-1)/2$ different classifiers are trained, with each training data from two different classes in the original training set, so that every class is pitted against each other class at least once. At prediction time, each binary classifier is applied to the sample x_m, and the class that had the highest number of predictions is voted to be the predicted class for that sample. In the case of a draw, the class is randomly assigned (Knerr et al. 1990).

3.3 Description of Data and Faults

The data in this study comes from a 3 MW direct-drive turbine which supplies power to a large manufacturing facility located near the coast in the South of Ireland. Data comes from the turbine SCADA system in the form of 10-minute operational and instantaneous alarm system data. The data covers an 11 month period from May 2014 - April 2015.

3.3.1 Operational Data

The turbine SCADA data contains 10-minute values for 61 different features. These include the average, max. and min. of wind speed, power and rotational speed, as well as the nacelle orientation and number of operating hours since commissioning. Also included in these features are over 30 temperatures, including inverter cabinet, blade and bearing temperatures. This data was used to train the classifiers and was labelled according to three different processes explained in Section 3.4.1. The operational data contains roughly 45,000 samples, representing the 12 months May 2014 - April 2015 analysed in this study.

3.3.2 Alarms Data

For this particular SCADA system, a new timestamped information, warning or alarm message is generated every time the operating state of the turbine changes, known as a "status" message. Thus, the turbine is assumed to be operating in that state until the next status is generated. Each turbine status has a “main status” and “sub-status” code associated with it. See Table 3.1 for a sample of the this alarm system data. Any main status code zero indicates that the turbine is generating or spinning up to generate, while anything above zero represents either faulty or curtailed operation, or that the system is operating close to the bounds of acceptable states (in these cases, the status counts as a "warning" status). Also included are times where the turbine is available to operate, but not producing power due to weather, grid or other events, e.g. status code 2 - “lack of wind”.

Table 3.1: WEC Status Data

Timestamp	Main	Sub	Description
13/07/2014 13:06:23	0	0	Turbine in Operation
14/07/2014 18:12:02	62	3	Feeding Fault: Zero Crossing Several Inverters
14/07/2014 18:12:19	80	21	Excitation Error: Overvoltage DC-link
14/07/2014 18:22:07	0	1	Turbine Starting
14/07/2014 18:23:38	0	0	Turbine in Operation
16/07/2014 04:06:47	2	1	Lack of Wind: Wind Speed too Low

3.3.3 Faults Classified

As mentioned in Section 3.3.2, any “main status” above zero indicates abnormal or potentially abnormal behaviour, but not necessarily a fault. Although over forty different types of faults occurred in the eleven months of data, only a small number occurred frequently enough to be able to train a classifier to recognise them. These faults are summarised in Table 3.2. Note that the fault frequency refers to specific instances of each fault, rather than the number of data points of operational data associated with it, e.g., a generator heating fault which lasted one hour would contain 6 operational data points, but would still count as one fault instance. In this table, *feeding faults* refer to faults in the power feeder cables of the turbine. These are found under the "power feeder cables" assembly of the "power module" sub-system in the Reder taxonomy in Appendix A. *Excitation errors* refer to problems with the generator excitation system, while *generator heating faults* refer to the generator overheating. These are both found in the "generator" assembly of the "power module" sub-system. Finally, *malfunction air cooling* refers to problems in the air circulation and internal temperature circulation in the turbine, found in the "cooling system" assembly of the "auxiliary system" sub-system.

Table 3.2: Frequently occurring faults, listed by status code, fault incidence frequency and number of corresponding 10-minute SCADA data points

Fault	Main Status	f	No. Pts.
Feeding Fault	62	92	251
Excitation Error	80	84	168
Malfunction Air Cooling	228	20	62
Generator Heating Fault	9	6	43

3.4 Methodology

In this case study, three levels of classification are attempted: fault detection, fault diagnosis and fault prediction. The general methodology for all three types of classification is shown in Figure 3.1. As can be seen, there are four main steps following a general machine learning process, described in detail in this section.

3.4.1 Data Labelling

The processes for labelling the data for each classification level are given below.

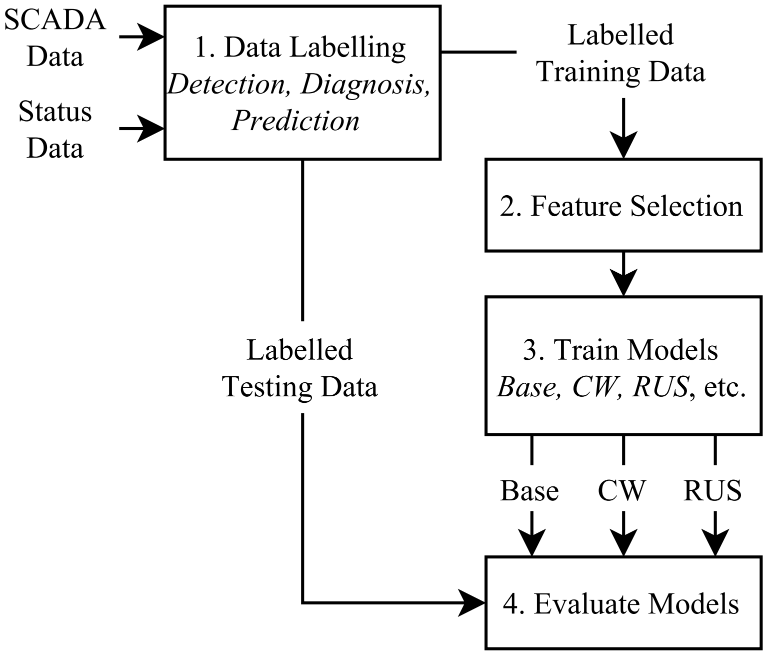


Figure 3.1: Methodology, following a typical machine learning approach. The abbreviations in step 3 are a selection of the approaches described in section 3.4.3

3.4.1.1 Fault Detection

The first level of classification is distinguishing between two classes: “fault” and “no-fault”. The fault data corresponds to times of operation under a set of specific faults mentioned in Section 3.3.3. For these faults, status messages with codes corresponding to the faults were selected. They were then labelled as follows:

$$y_t = \begin{cases} \text{F}, & \text{if } t_s - 600s < t \leq t_e + 600s \\ \text{O}, & \text{otherwise}, \end{cases} \tag{3.15}$$

where y_t is the label at time t, t_s is the start of the faulty state and t_e is the end of the faulty state (i.e. the start of the next non-fault status in the data). A time band of 600s before the start, and after the end, of these turbine states was used to match up the associated 10-minute operational data. This was selected so as to definitely capture any 10-minute period where a fault occurred, e.g., if a power feeding fault occurred from 11:49-13:52, this would ensure the 11:40-11:50 and 13:50-14:00 operational data points were labelled as faults. No matter the type of fault, all faults were simply labelled as the “F” class. All the remaining data in the operational dataset was then given the label “O”, representing all other data. This is because the remaining data did not necessarily represent fault-free data; it just meant it did not contain the faults mentioned in Section 3.3.3, but could have included other, less frequent faults or times when the turbine power output was being curtailed for any one of a number of reasons mentioned previously.

3.4.1.2 Fault Diagnosis

Fault diagnosis represents a more advanced level of classification than simply fault detection. The aim of fault diagnosis is to identify specific faults from the rest of the data. Faults were labelled in the same way as in the previous section, but this time, each fault was given its own specific label. Again a time band of 600s before the start and after the end of each fault status was used to match up corresponding 10-minute operational data. Any data that remained, i.e., was not labelled as one of the faults mentioned in Section 3.3.3, was again given the “O” label, for a total of five different classes (the four fault classes as well as the “O” class). If two faults occurred concurrently, then the data point was duplicated, with each point having a different label.

3.4.1.3 Fault Prognosis

Fault prognosis represents an even more advanced level of classification than fault diagnosis. The aim of this level of classification was to see if it was possible to identify that a specific fault was imminent from the full set of operational data. After initial tests, it was decided to focus prediction only on generator heating and excitation faults as these showed the greatest promise for early detection. Details of this can be found in Section 3.5. The other faults were included in the “O” label along with the rest of the data, for a total of three classes.

For fault prediction, the times during which the turbine was in faulty operation were not labelled as such. Instead, operational data points leading up to each fault were labelled as “PF”, meaning pre-fault, for each specific fault, as follows:

$$y_t = \begin{cases} \text{PF}, & \text{if } t_s - 600s - w_1 < t \leq t_s - 600s - w_2 \\ \text{O}, & \text{otherwise}, \end{cases} \tag{3.16}$$

Table 3.3: Values of w_1 and w_2, in hours, used for fault prediction

Case	w_1 (hours)	w_2 (hours)
Case 1	1	0
Case 2	2	1
Case 3	3	2
Case 4	5	2
Case 5	12	2
Case 6	24	12

where $\{w_1, w_2; w_1 > w_2\} \in \mathbb{N}^+$ are windows of time in advance of the fault. In this way, a window of time w_1 preceding the fault is labelled as pre-fault data, while leaving a window w_2 as a "minimum prediction time" where a useful warning can be given before the fault occurs. The time between $t_s - 600s - w_1$ and $t_s - 600s - w_2$ is known as the "pre-fault window".

A number of separate cases representing different values of w_1 and w_2 were tried to see how far in advance an accurate prediction could be made. These can be seen in Table 3.3. For example, for case 6, all data points with timestamps between 24 and 12 hours before a generator heating fault occurred were labelled as "PF - generator heating". The same was applied to excitation faults. All remaining unlabelled data points were labelled "O". Once again, if different faults occurred concurrently, the data points were duplicated and given different labels.

The time spanned by w_1 and w_2 can be thought of as a wide RUL estimate for a particular fault. For example, under case 6, if a live sample is classified as "PF - generator heating fault", the RUL for that assembly is between 12 and 24 hours. In this case, the minimum PH is 12 hours. The technician or maintenance operator would then have between 12 and 24 hours between this point being detected and the fault actually occurring to remotely or manually inspect the generator and organise any necessary maintenance actions ahead of time. If maintenance is needed, this can reduce the logistics lead time or allow it to be scheduled in conjunction with other maintenance activities for maximum economic benefit.

3.4.2 Feature Selection

The full operational dataset had more than sixty features, many of which were redundant, incorrect or irrelevant. Because of this, only a subset of specific features were chosen to be included for training purposes. It was found that a number of the original features corresponded to sensors on the turbine which were broken, e.g., they had frozen or blatantly incorrect values, while others contained duplicate or redundant values. These were removed. A number of the remaining features which were deemed as obviously irrelevant based on domain knowledge were also excluded, e.g. features relating to the cumulative uptime of the anemometer or the open/closed state of the tower base door. This resulted in 39 remaining features. A subset of *these*, corresponding to 12 temperature sensors on the inverter cabinets in the turbine, all had very similar readings. Because of this, it was decided to instead consolidate these and use the average and standard deviation of the 12 inverter temperatures. This resulted in 29 features being used to train the SVMs. It was decided to scale all features individually to unit norm because some, e.g., power output, had massive ranges from zero to thousands, whereas others, e.g., temperature, ranged from zero to only a few tens.

3.4.3 Model Selection

Each of the labelled datasets mentioned in section 3.4.1 were randomly shuffled and split into training and testing sets, with 80% being used for training and the remaining 20% reserved for testing. Fault data inherently has a massive imbalance in the distribution of samples in each class, in that the number of fault-free data massively outnumbers the other classes. This case study was no exception - the number of fault-free samples was on the order of 10^2 times that of fault samples for all three levels of classification. This can sometimes be a problem for SVMs, so a number of different approaches were taken to address the issue. These included adding an extra "class weight" hyper-parameter, or under/oversampling the training data being fed into the SVM. In addition to this, because fault diagnosis and prediction represented a greater classification challenge, approaches using ensemble meta-learners were used to reduce bias and variance in the results. In any case, the test data was not altered in any way so as to preserve the imbalanced distribution seen in the real world.

The various approaches used can be broken into four general categories: "ge-

Table 3.4: Summary of different training approaches taken, showing their general category and which classification levels used these approaches

Approach	Category	Classification Levels
Base	General	Det/Diag/Pred
CW	General	Det/Diag/Pred
RUS	Undersample	Det/Diag/Pred
CC	Undersample	Det/Diag/Pred
TL	Undersample	Diag/Pred
ENN	Undersample	Diag/Pred
SM	Oversample	Diag/Pred
EE	Ensemble	Diag/Pred
Bag-CW	Ensemble	Pred
Bag-RUS	Ensemble	Pred

neral", "undersampling", "oversampling" and "ensemble methods". Because of the simpler problem it posed, fault detection was only carried out using a small subset of these methods, whereas both diagnosis and prediction were carried out using additional approaches, summarised in table 3.4. Additionally, fault prediction used two bagging-based approaches. These are described in detail in the following section.

Note that each level of classification above fault detection used multi-class classification based on the “one-against-one” approach (Knerr et al. 1990). For all three levels of classification, the models were trained using scikit-Learn’s implementation of LibSVM (Chang & Lin 2011, Pedregosa et al. 2012). For the fault prediction case, all approaches were initially trained based on the labelled dataset representing fault prediction Case 1 from Table 3.3, as described in Section 3.4.1.3. The best performing approach was then used to train classifiers on cases 2-6.

3.4.3.1 General Approaches

Base Case (Base) In the base case, i.e. "vanilla" SVM, a randomised grid search was performed over a number of hyperparameters to find the ones which yielded the best results on the full set of training data. These were then verified using 10-fold cross validation. The scoring metric used for cross validation was the $F1$ score (see eq. 2.4). The hyperparameters searched over were C, which controls the number of samples allowed to be misclassified, ϵ which defines how much influence an individual training sample has, and the kernel used. The three kernels which were tried were the simple linear kernel (i.e. $K(x_m, x_i) = x_m^T x_i$), the polynomial kernel (eq. 3.13) and the radial-basis (Gaussian) kernel (eq. 3.14).

The training data from all undersampling and oversampling methods were fed into an SVM following this approach. Additionally, the meta-learners using the ensemble methods also followed this approach.

Addition of Class Weight (CW) In this approach, an additional hyperparameter, the class weight, $c.w.$, is added during training. This is a weighted scaling factor used when calculating C for the minority class. The new value for C for the fault class, C_w, is calculated as in Eq. 3.17.

$$C_w = C * c.w. \tag{3.17}$$

Training is then performed as in the base case. A number of different class weights ranging from 1 ($C_w = C$) to 1,000 ($C_w = 1000 * C$) were added to the set of hyperparameters being searched over for this approach. There is no over- or undersampling used in this method.

3.4.3.2 Undersampling Methods

Random Undersampling (RUS) This approach randomly under-samples the majority fault-free class (without replacement), so that the number of fault-free samples in the training data was equal to the number of fault samples.

Cluster Centroids (CC) This undersampling method splits all the samples of the majority class into k clusters using the k-means algorithm. The centroids of these clusters are then used as the new samples for this class. In this case, the value of k used was equal to the number of samples in the minority class.

TomekLinks (TL) TomekLinks is a modification of the condensed nearest neighbour algorithm which undersamples from the majority class by eliminating samples which are close to the decision boundary between the two classes (Tomek 1976). For this application, the fault free class was undersampled to bring the number of samples down to near the number of samples in the largest fault class.

Edited Nearest Neighbours (ENN) The Edited Nearest Neighbours method is a slight modification of the k-nearest neighbours method used to under-sample from the majority class (Wilson 1972). This is shown in alg. 1 (Wilson 1972). In this way, the fault-free class was significantly reduced in size.

Algorithm 1: ENN Under-sampling

```
Data: Samples $X = (x_1, ..., x_n)$, labels $Y = (y_1, ..., y_n)$
Result: Reduced set of samples $X'$ and labels $Y'$
Set $X' = X$
Set $Y' = Y$ for each sample $x_i$ in $X$ do
    Find the $K$-nearest neighbours to $x_i$ among $\{x_1, x_2, ..., x_{i-1}, x_{i+1}, ..., x_n\}$
    Find the class $y'$ associated with the largest number of points among the
      $K$-nearest neighbours, with ties being settled at random
    If $y_i \neq y'$, remove $x_i$ and $y_i$ from $X'$ and $Y'$, respectively (i.e.
      $Y' = Y \setminus \{y_i\}; \quad X' = X \setminus \{x_i\}$
```

3.4.3.3 Oversampling

SMOTE (SM) SMOTE (Synthetic Minority Over-Sampling Technique) is an algorithm that generates synthetic samples for the minority class. New samples are generated along the line connecting each sample in the minority class to its k-nearest neighbours (Chawla et al. 2002). SMOTE uses five nearest neighbours. In this case, a number of new synthetic samples were generated for each fault class to bring the number of samples in line with the number of fault-free samples.

3.4.3.4 Ensemble Learners

Bagging (Bag-CW, Bag-RUS)

Bagging, or BootstrapAggregating, is an ensemble technique designed to reduce overall variance and avoid over-fitting (Breiman 1996). It creates multiple smaller training sets by sampling from the full training set. It then builds a classifier for each of these subsets (in this case an SVM). Each classifier in the resulting ensemble then votes with equal weight to give the predicted class of each new sample. Two different bagging classifiers were trained; one using the additional "class weight" hyperparameter, as described in section 3.4.3.1; and another using a randomly undersampled training set, as described in section 3.4.3.2.

EasyEnsemble (EE)

EasyEnsemble builds upon the AdaBoost algorithm to address the problems of class imbalance (Liu et al. 2006). AdaBoost takes the output of a number of "weak learners" trained on subsets of the full dataset and assigns weights to each (Freund & Schapire 1995). The weighted sum of these learners is the output of the AdaBoosted classifier. With EasyEnsemble, the subsets of the training data are selected by undersampling the majority class, in a similar way to that described in section 3.4.3.2, so that in each subset the number of samples in the minority and majority classes are equal.

3.4.4 Model Evaluation

A number of scoring metrics were used to evaluate final performance on the test sets for fault detection and fault diagnosis, as well as the six test sets representing cases 1-6 for fault prediction. A high number of false positives can lead to unnecessary checks or corrections carried out on the turbine, and this was captured with the precision score, where a higher score represents a lower false positive rate (eq. 2.2). A high number of false negatives, on the other hand, can lead to failure of the component with no detection having taken place (Saxena et al. 2008). This is captured by the recall score, where a higher number indicates a low ratio of false negatives (eq. 2.3). The F1-Score was also used, which is the harmonic mean of precision and recall (eq. 2.4). Confusion matrices were used where appropriate to give a visual overview of performance and show absolute numbers (figure 2.21). The overall accuracy of the classifier on each test set was not used as a metric for the reasons outlined in section 2.8.4.3.

Specificity was deemed to not provide any additional useful information compared to the recall, but was used in one specific case for benchmarking against specificity scores in a previous study.

3.5 Results & Discussion

3.5.1 Fault Detection

For fault detection, the recall score was generally high, ranging from .78 to .95, as seen in Table 3.5. However, the $F1$ score was brought down by poor precision scores all-round, representing a degree of false positives. Specificity was highest using the CW method with a score of .82, followed by RUS with .67, but suffered on the Base and CC methods, with .49 and .5, respectively. The best balanced performance was seen on CW, with good recall and specificity scores of .83 and .82, respectively. The precision of .04 was very low, but was among the highest of all results seen. The scores were an improvement on those obtained in (Kusiak & Li 2011), where the best recall and specificity for fault detection were 0.84 and 0.66, respectively (compared with .95 and .82 here).

Table 3.5: Results for Fault Detection

Method Used	Pre.	Rec.	F1	Spec.
Base	.02	.78	.04	.49
CW	.04	.83	.07	.82
RUS	.02	.9	.04	.67
CC	.02	.95	.03	.5

3.5.2 Fault Diagnosis

The scores on each fault for every fault diagnosis method are summarised in Figure 3.2. As can be seen, scores for the SVM trained using the CW method were generally slightly worse than RUS, apart from in the case of aircooling faults with a recall score of 0.7 (up from the randomly undersampled training set score of 0.33). This increase, however, may be because there were only 7 instances of air cooling fault in the test set, leaving it open to different test scores in each case. The CC and EE methods both performed slightly worse again, with CC slightly beating EasyEnsemble. The SVM trained on data undersampled using the ENN approach performed worse than both the EE and CC methods overall, but achieved a better F1 score on generator heating faults than the "vanilla" RUS method (0.82 vs. 0.8). However, this was down to improved precision (0.88 vs. 0.73) at the expense of recall (0.78 vs. 0.89). Both TL and SM performed by far the worst overall. SM uses synthetic data to populate the minority class, so its poor performance suggests that using synthetic data is not suitable for this application.

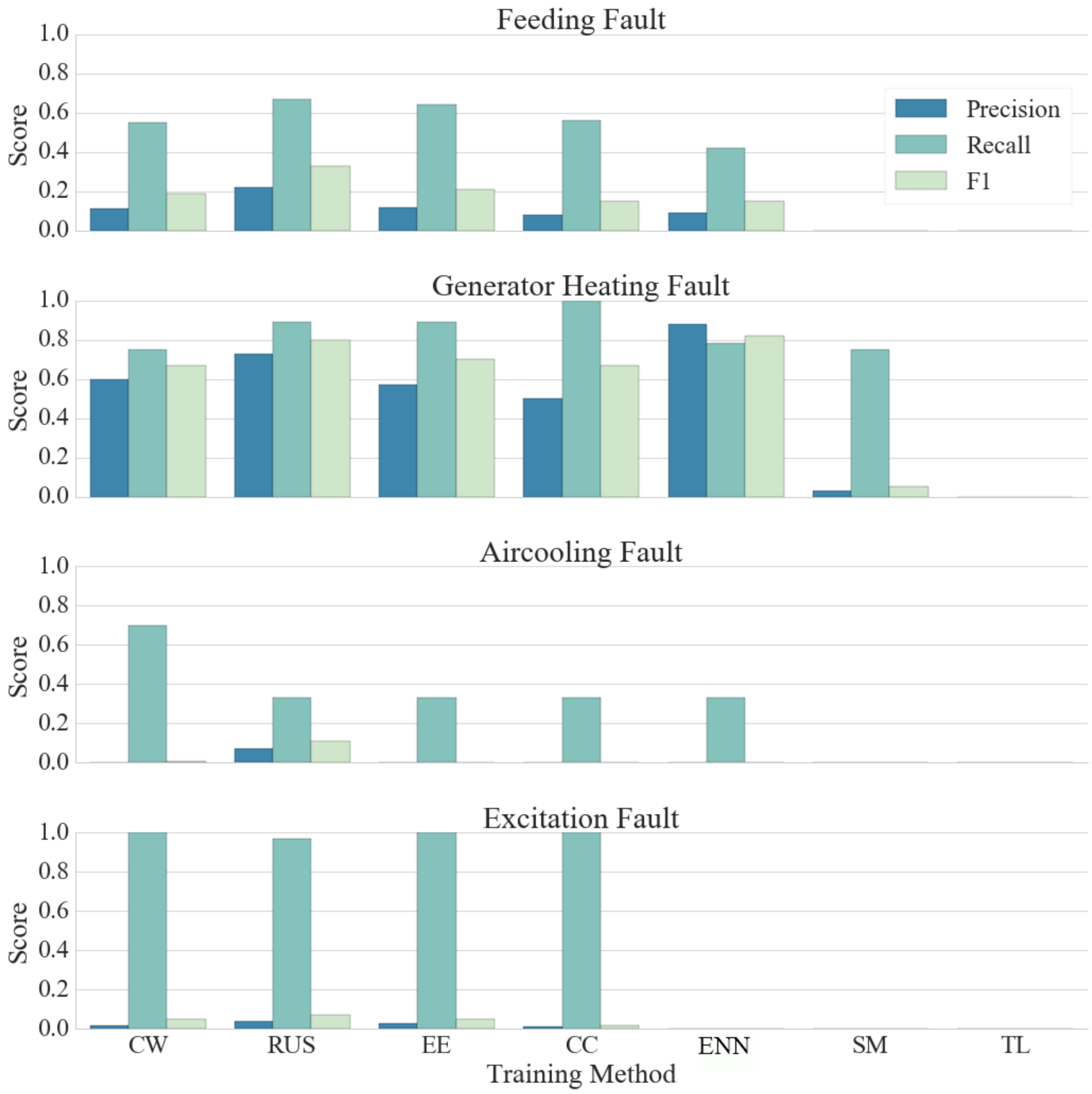


Figure 3.2: Precision, Recall and F1 scores for fault diagnosis across various training methods

The RUS method performed the best overall. Results for this can be seen in the confusion matrix in Figure 3.3. Generator heating faults showed a low proportion of false positives, as well as correctly catching 89% of faults. Excitation faults similarly showed a high proportion of caught faults at 97%, but was let down by a high number of false positives leading to a low precision score of .04. 67% of feeding faults were caught, but here also the number of false positives led to a precision score of .22.

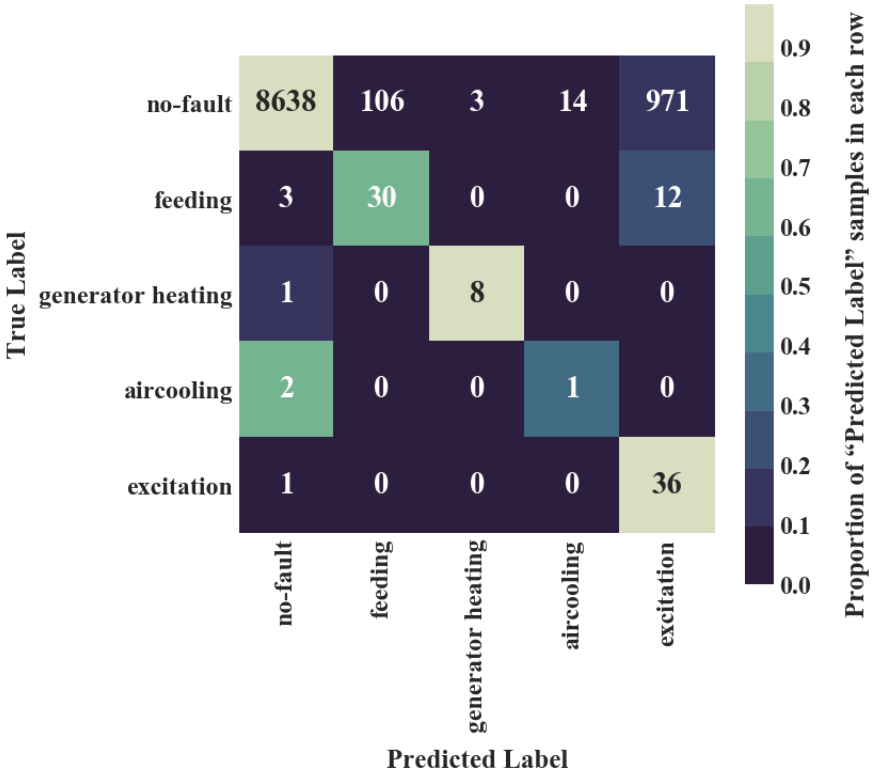


Figure 3.3: Confusion Matrix for fault diagnosis using the RUS training method. This shows the absolute number of correctly or incorrectly labelled samples, as well as the proportion of "Predicted Label" samples in each "True Label" row.

3.5.3 Fault Prediction

Early results on fault prediction showed that generator heating and excitation faults showed the best promise for effective prediction. Feeding and air cooling faults showed very poor performance, possibly due to a separating hyperplane being hard to find in the limited data available for these particular faults. For this reason, it was decided to focus on generator heating and excitation faults.

As described in Section 3.4.3, the various training methods were first tried on the labelled dataset representing Case 1 from Table 3.3. The best performing of these was then used on the other fault prediction windows.

The scores for each training method across the different faults using prediction Case 1 are shown in Figure 3.4. Surprisingly, the best test scores were not seen on any of the ensemble classifiers, but on the SVM trained with the CW method, using a linear kernel. The full results for CW can be seen in Table 3.6. As can be seen, the recall score is very good, but again the $F1$ is brought down by poor precision. RUS came in close behind, but with lower precision on both faults. EE and CC both performed worse, with lower scores on precision for generator heating faults. ENN and SM both performed better than CW on generator heating faults, but were let down by a zero $F1$ score on excitation faults. TL performed very badly, with $F1$ scores of zero for both faults. Both bagging methods also performed very poorly.

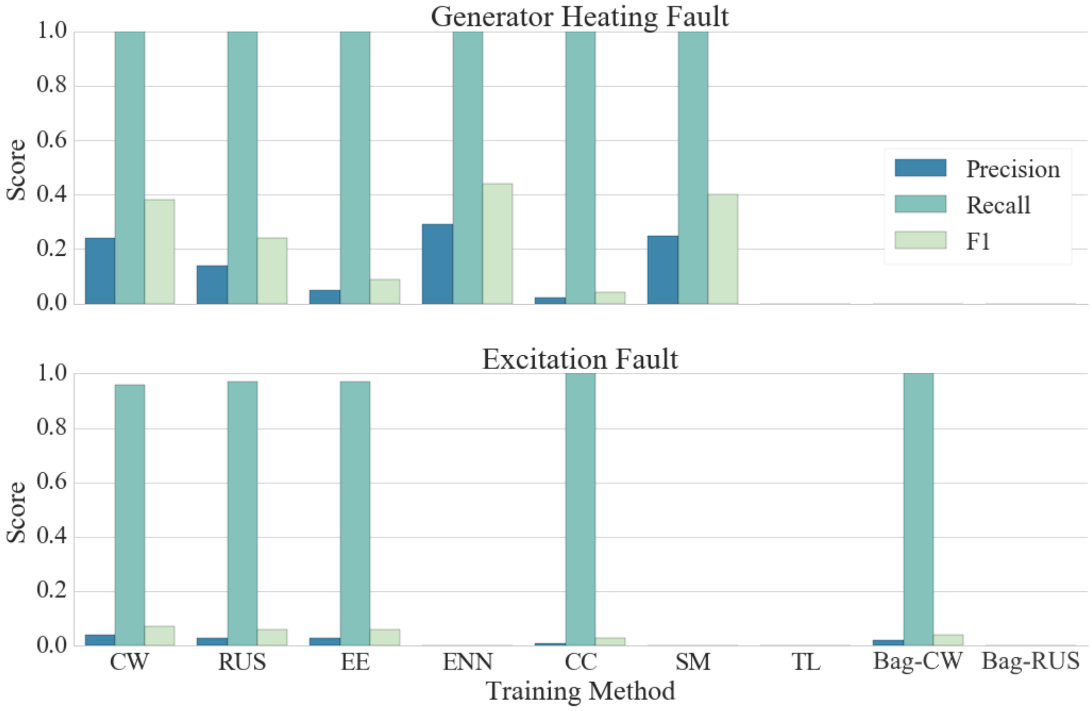


Figure 3.4: Precision, Recall and F1 scores on fault prediction for various training methods

Table 3.6: Results for Fault Prediction Using CW Method on Fault Prediction Case 1

Faults	Pre.	Rec.	F1
Generator Heating Fault	0.24	.98	0.38
Excitation Fault	0.04	0.96	0.07

The test results from CW for various cases of time in advance of a fault, as described in Section 3.4.1.3, can be seen in Figure 3.5. The recall score for both generator heating and excitation faults stays relatively high for all cases. Both have a recall score of .97-.99 for cases 1 & 2. This falls to around 0.8 for cases 3 & 4 for generator heating faults, but rises again to above 0.9 for cases 5 & 6. Excitation faults start to drop just below 0.8 for cases 5 and 6, but this is still quite high. These results show that good indicators of a developing fault are seen up to 12-24 hours in advance of a fault solely looking at 10-minute SCADA data. Previous work done in (Kusiak & Li 2011) showed a recall score of just .24 one hour in advance of a specific fault, and this was the furthest window tested. It should also be noted that the results in that work represented a test set which did not fully sample the fault-free class. In (Kusiak & Verma 2011), the maximum prediction time was 10 minutes, with a recall score of .71 when detecting blade pitch faults, using SCADA data at a 1s resolution (rather than the 10 minute data used in this study). Hence, these results are extremely promising.

Confusion matrices for cases 1 and 4 are shown in Figure 3.6. For excitation faults, although nearly all faults were successfully predicted in advance, there were a high number of false positives, leading to a low precision score of below 10% in both cases. Generator heating faults saw more success. There was a 40% precision score for case 1, and a 22% score for case 4. Although these scores are quite low, the inherent value of a SCADA-based system is that it does not require the installation of any additional sensors or other hardware, so can sit alongside existing systems providing additional CM functionality with little additional cost. Any alarms generated by the system showing impending faults can be remotely investigated to determine if action needs to be taken. An extension to this work can be found in (Hu et al. 2016), which shows promising early results in improving precision scores for fault detection by using time-lagged and statistical features.

3.6 Conclusion

Various classification techniques based on the use of SVMs based on SCADA data to provide additional CM functionality to wind turbines were investigated. All three levels of CM, as described in section 2.1.4.1, were attempted: fault detection, i.e. distinguishing between faulty and fault-free operation; fault diagnosis, whereby faulty operation was identified and subsequently the nature of the fault diagnosed; and, fault prediction, where a rough RUL estimate was given for a particular type of fault. The classification techniques employed various different ways of training SVMs to deal with the problems of imbalanced data, including re-sampling the data as well as using ensembles of SVMs.

The results were promising and show that fault detection is possible with very good recall and specificity, but the F1 score is brought down by poor precision. In general, this was also the case for diagnosing a specific fault. More impor-

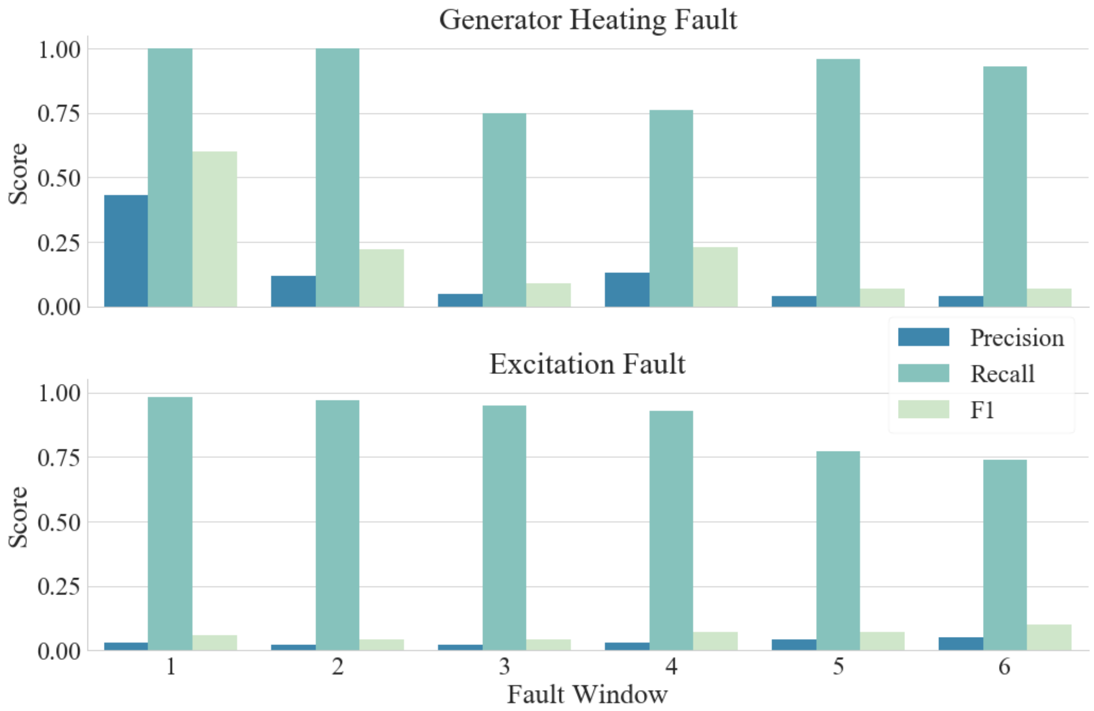


Figure 3.5: Precision, Recall and F1 scores for using CW training method to predict generator heating and excitation faults for the cases shown in Table 3.3

tantly, predicting certain types of faults was possible up to 24 hours in advance of the fault with very high recall scores. Although the F1 score was here again brought down by poor precision, this still represents a significant increase over what was previously possible using 10 minute SCADA data, and improving the precision scores would represent a very important step forward in being able to rely on SCADA data for accurate fault prediction. In all cases, using the CW or RUS methods to deal with imbalanced data provided the best overall performance. These methods are much simpler to implement that comparable under- or over-sampling methods.

The results also demonstrated one of the advantages of a classification-based approach over other approaches for CM based on SCADA data: a single SVM model can be trained to diagnose or predict a number of different faults. In this case, some success was demonstrated in diagnosing and predicting faults across a range of assemblies from the generator to the power feeder cables. This is in contrast to, for example, NBMs, which rely on a model being built to represent parameters such as temperatures associated with specific components.

The data used in this study related to a single turbine over an eleven month period. This represents a limitation in what can be achieved; with additio-

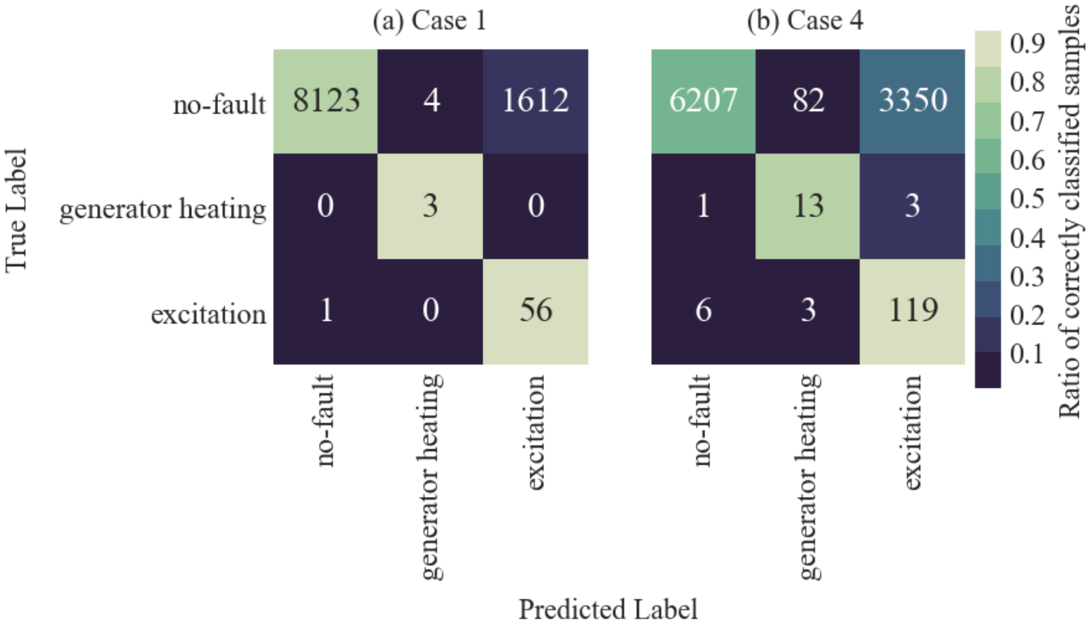


Figure 3.6: Confusion matrices showing the ratio of correctly classified samples for fault prediction using the CW training method on prediction cases 1 and 4

nal data, i.e., from more turbines over a longer period, better fault prediction should be possible due to more positive examples being available for training. As well as this, advanced feature extraction and selection would enable even higher scores. An extension to this work in (Hu et al. 2016) showed improved precision scores on fault/no-fault classification by using domain knowledge, temporal and statistical features, followed by using feature selection methods to find only the relevant features and speed up training time.

While this work used the alarm system directly for labelling and describing faults, it did not take into account multiple concurrent alarms appearing during fault events. The available data also provided no way to assess whether the alarms were an accurate representation of the faults being experienced. While this approach is consistent with other work in the literature, it poses some potential issues which will be discussed in the following chapter, chapter 4. A solution to these labelling issues based on sequences of alarms will be proposed, before a formalised framework for using classification as a CM tool will be described in chapter 5.

The work presented here also relied solely on classification metrics for evaluating the system. This is also consistent with what is seen in literature, but fails to take into account how alarms based on such a system would be implemented and how they would perform. This will also be addressed in chapter 5.

3.6.1 Research Objectives

ROs 1 & 2 of this thesis were:

RO1. Determine what level of condition monitoring can be performed using classification techniques

RO2. Investigate different techniques for dealing with classification based on imbalanced datasets and evaluate their suitability for fault detection, diagnosis and prediction

It was shown that all three levels of CM are possible, though the number of false positives was quite high. Hence, all three levels of CM are possible, but further work is needed to bring up the precision score. Although this meets RO1, this will be further investigated in subsequent chapters.

Different techniques for dealing with imbalanced data were also presented, compared and evaluated across all three levels of CM. This showed that the addition of a class weight, or simply undersampling the majority class, were the best ways of dealing with class imbalance in the training stage, satisfying RO2.

Chapter 4

A More Granular Picture of Historical Failures

4.1 Introduction

A number of techniques for CM based on SCADA data discussed in section 2.8 depend on accurately determining the historical state of a turbine. For NBM, this means determining the times when the turbine was operating nominally and fault free, as opposed to needing to explicitly label when specific faults took place. In works such as (Kusiak & Verma 2013, Bangalore et al. 2017, Zhao et al. 2017), the authors use clustering to identify these periods of normal operation by assuming abnormal data will lie in distinct outlying clusters. In others such as (Butler et al. 2013, Park et al. 2014), outliers are filtered out with the assumption that for the majority of normal operation, parameters such as power, wind speed, or certain temperatures, should lie within certain statistical bounds. For classification, because classifiers also learn what the actual faults themselves look like, more granularity is needed. Specific types of faults must be labelled as such, and the small number of fault samples mean any mislabelled sample can disproportionately affect overall performance. If quality historical data on operational states and failures can be obtained, this can also be useful for assessing the reliabilities of assemblies and turbine models in an operator's fleet.

4.1.1 Ideal Case

Ideally, this data would come from a structured stoppage and failure database for the wind farm, containing information such as the root cause or failure mode (if available), accurate timestamps for when the turbine stopped, the duration of the down time, and when (if any) maintenance occurred (Peter Tavner 2012). Here, "structured" refers to maintenance entries being selected from a prescribed list and mapped to a standardised turbine taxonomy showing the component or sub-system affected, such as those mentioned in section 2.4. This is opposed to "unstructured" data such as comments explaining what work was done, which cannot be easily automatically searched and categorised. It would also be useful to have some way of identifying periods of downtime in the data related to events such as grid faults or noise or shadow-related curtailment, so that these can potentially be taken into account. For classification purposes, this means the 10-minute operational data can be automatically and accurately labelled with each of these periods, and a classifier could be trained on historical data to distinguish between normal operation and when a particular type of fault is developing. In addition, this would allow a comprehensive picture of turbine reliability to be built down to the component level, giving operators detailed information for all makes and models of turbines in their fleet (Reder et al. 2016). However, as was seen with the reliability studies in section 2.6, obtaining accurate fault data can be challenging. This is particularly true for researchers, as noted in (Tautz-Weinert & Watson 2017*a*, Kusiak 2016, Wennerhag & Tjernberg 2012). The two main ways of obtaining failure data seen in the literature for both reliability studies and SCADA-based CM have been from fault logs and the SCADA alarm system.

4.1.2 Labelling by Fault Logs

Maintenance records can be an effective way of determining the historical operational history of a turbine, as was done for the classification in (Godwin & Matthews 2013). However, fault and maintenance logs are not always stored in structured, or even digital, databases, and can vary across different OEMs, operators and, in some cases, even individual maintenance technicians. Field records can be incomplete, inaccurate or lacking in sufficient detail (Guo et al. 2009). Furthermore, maintenance logs do not necessarily capture instances when the turbine was not operating for non-fault reasons (such as previously mentioned grid, noise or shadow-related stoppages or curtailment). A recent report by the European Academy of Wind Energy (EAWE) identified the lack of standardised maintenance reporting as a long-term research challenge in the wind industry (van Kuik et al. 2016). This is echoed by IEA Wind Task 33, which aims to provide recommendations for standardised failure and reliability reporting (Hahn 2017).

This is an issue which has not seen a lot of dedicated coverage in the literature, as noted by the authors in (Tautz-Weinert & Watson 2017*a*). Here, a review of maintenance logs for four different wind farms, containing 210 turbine-years worth of data, was performed. The best-case of all of these recorded a list of all stoppages in the period the records covered. However, even in this case, comments giving context to the stoppages were only added for major component replacements, or occasionally giving other reasons for the stoppage. Furthermore, for major replacements, no mention was given as to whether the work was preventive or corrective (i.e. whether the result of a full failure). The given timestamps were also only for when the repair team arrived on site as opposed to when the turbine failed (assuming this was corrective rather than preventive maintenance). For two other farms in the study, only a list of major component replacements were recorded, again not noting the times of actual failure. For the final farm, no service record at all was available.

These issues are echoed in parts of other publications. For example, the authors of (Kaidis et al. 2015) note that maintenance logs can be difficult to read or incomplete, or only available in hard copies. This sentiment is also reflected in a Sandia National Laboratories report in (Hill et al. 2009), where it is noted that work orders may list the amount of time for a repair operation, but not the actual times the turbine was down. These issues make it difficult to accurately label when faults occurred at the granularity needed. Furthermore, even if fault logs are accurate, if the data is not stored in a structured way, or relies on comments to contextualise the stoppage or fault event, labelling the data for classification can be a tedious and involved process. All of these findings are consistent with this author's own experience - data available from two different data sets contained information on major repairs, but this was stored in PDF documents or as comments on spreadsheets.

4.1.3 Labelling by the Alarm System

Another way of gleaning accurate fault information is from the turbine's alarm system, as done by the classification in (Hu et al. 2016, Kusiak & Li 2011, Chen et al. 2013). This has the advantage of being consistent and automatically recorded with accurate timestamps, regardless of operator maintenance policies or technician error. Therefore, if specific types and times of faults, scheduled maintenance, or other types of stoppages can be interpreted from this data, the logic can be applied to all turbines which use a similar SCADA alarm system.

As explained in section 2.5, fault alarms are usually triggered when the turbine controller detects an operating condition that falls outside of defined acceptable bounds, such as the rotor exceeding a certain speed, or the bearing exceeding a certain temperature. Hence, the presence of a turbine alarm does not always indicate that a fault has taken place - they can be triggered as a precautionary measure to avoid damage. This can make it difficult to perform classification as the leading fault signatures for these less severe situations may not be as obvious in the data. By giving criteria for what constitutes a fault in a similar way to some of the reliability studies discussed in section 2.6, e.g. related to the duration of a stoppage or whether repairs took place, this problem may be avoided.

Some turbine alarm systems have functionality whereby a number of alarms, warnings or information messages can be active at any one time, or in very quick succession. Turbine alarms also often occur in quantities too large for effective analysis, as will be discussed in chapter 6 (Yang et al. 2014, Qiu et al. 2012). This is particularly true during fault events, where alarms occur in "alarm showers". These qualities can make it difficult to discern the precise reason for or times of a particular stoppage. Some approaches have been taken in the past to try and address this; for example, in (Kusiak & Li 2011), the authors assign severity categories to the fault alarms. If there are a number of concurrent fault alarms, the authors assign the fault to the most severe alarm which occurred, but no linking of subsequent alarms is made, or an attempt to quantify faults in terms of duration of the associated stoppage. In (Kusiak & Verma 2011), meanwhile, the authors linked particular types of faults to particular sequences of alarms. Once again, however, the duration or severity of faults were not taken into consideration, and the sequences of alarms were not taken into account when labelling the data. In the previous chapter, a case study on CM using a classification-based approach was performed using alarm system data to label the faults. The alarm system communicated what "status" the turbine was operating in, and it could only operate in a single status at a time. Hence, the issue of concurrent alarms was avoided. However, the duration or severity of faults, or whether particular sequences of alarms had any significance, was not taken into consideration.

4.1.4 Solution

If some way of automatically mapping alarm sequences as they occur during stoppages to specific types of system downtime or faults can be found, much of the functionality of the "ideal" maintenance database described previously can be captured, without the need for manual interpretation. Because the information would be recorded in a standardised and structured way, this would allow for automatic labelling of SCADA data, avoiding a tedious manual step. In this chapter, a method for building a database of historical turbine stoppages from SCADA alarms and availability data is proposed. Each stoppage will contain information on the affected assembly (or whether it was due to grid or weather-related issues), the duration the turbine was down for, and whether it resulted in any on-site intervention by technicians (i.e. fault levels 3 and 4 from section 2.5). This will allow more sophisticated labelling of data than the process used in chapter 3, and will form part of a proposed framework described in chapter 5. Section 4.2 will describe the data used to demonstrate the developed methodology. Section 4.3 will then describe the methodology. Section 4.4 will describe the results, and finally section 4.5 will draw conclusions.

4.2 Description of Data

The data used to demonstrate the methodology in this chapter comes from a site in the East of Ireland with complex terrain consisting of eleven 2.5 MW DFIG turbines. The data covers November 2015 - April 2016 and is composed of three parts: SCADA operational data, availability data and alarms data. There were over 260,000 10-minute SCADA data samples spread across all turbines, with each sample having over 90 different parameters, and an equal number of availability samples. The availability data tracked the availability states of the turbine over every 10-minute period by counting how many seconds in each period the turbine was in one of the following states:

Table 4.1: Ten Minute Availability Data

TimeStamp	OK	Down	Grid	Weath	Maint	Rep
09/06/2015 14:10:00	600	0	0	0	0	0
09/06/2015 14:20:00	400	200	0	0	0	0
09/06/2015 14:30:00	0	600	0	0	0	0
09/06/2015 14:40:00	100	500	0	0	0	0
09/06/2015 14:50:00	600	0	0	0	0	0

- OK - the turbine was operating normally
- Down - the turbine was not operating due to a fault detected in one of its subsystems
- Grid - the turbine was not operating due to a grid fault
- Weath - the turbine was not operating, or was curtailed, because of severe weather
- Maint - the turbine was down for routine/scheduled maintenance
- Rep - the turbine was down for unplanned repairs. This also references any time a technician accessed the turbine for a simple manual restart.

A sample of this data can be seen in table 4.1.

The alarm system was similar to the general system described in section 2.2. A sample of this can be seen in table 4.2. Note the codes have been randomised and descriptions heavily edited for anonymity purposes. Note also that the fault alarms for the turbine models used in this study are further split into "fault" and "critical fault" categories, with "critical fault" alarms requiring at least a manual on-site reset to restart the turbine (level 3 from section 2.5), while "fault" alarms cause the turbine to shut down, but in some cases can be remotely reset (level 2 from section 2.5). The OEM-assigned categories included references to various assemblies within the turbine, e.g. "pitch" or "generator", as well as assorted other categories such as "no fault", "weather" and "grid". Altogether, there were over 100,000 alarm instances in the dataset, with 232 unique alarm codes.

Finally, maintenance logs in the form of spreadsheet and PDF documents were available to evaluate the accuracy of the stoppage database created.

Table 4.2: Sample alarm system data from a certain day

t_s	t_e	Code	Description	Category	Severity
02:13:38	07:56:14	a_{41}	Normal Operation	No Fault	Information
07:56:14	08:37:32	a_{91}	Low wind cut out	Weather	Information
08:37:32	08:39:21	a_{21}	Start Up	No Fault	Information
08:39:21	23:44:02	a_{22}	Normal Operation	No Fault	Information
23:44:02	17:22:18	a_{122}	Pitch Thyristor Fault	Pitch	Fault

4.3 Methodology

As mentioned previously, every time the turbine shuts down or production is curtailed for any reason, a number of alarms are generated by the turbine's control system. Note that, "alarms" here refers to both warning/fault alarms as well as information messages. The gap between the first alarm(s) appearing which signify the turbine stopping/being curtailed, and the turbine coming back on-line, can be anything from minutes to weeks, depending on the reason for the stoppage. Because many alarms are generated both instantaneously when the turbine stops, as well as over the duration of the stoppage as reactions to these other alarms, it can be difficult to assign a reason for the stoppage from any one alarm. This can lead to errors in labelling SCADA data for classification.

To avoid this problem, it is instead possible to (i) identify *batches* of alarm sequences as they occur during each stoppage, and (ii) use a specific set of rules to attribute each batch to a stop category. These two steps will be described over the following sections.

At this point it is useful to explain some notation and terminology that will be used going forward. The word "alarm" in this work refers to a single type of warning/fault alarm or information message, and its associated code, which can be triggered by the control system. The alarms will be labelled with a_c, where c is the code for that alarm (note again that these codes have been randomised for confidentiality reasons). For example a_{23} refers to alarm code 23, with the associated OEM-assigned description, category and severity, as described in section 4.2. The full set of all k alarms is called A:

$$A = [a_1, \dots, a_k] \tag{4.1}$$

Alarm *instances*, on the other hand, refer to individual instances of specific alarms/information messages with a start time, t_s, and end time, t_e. The full set of alarm instances will be labelled L, and specific individual instances l. In this way, all alarm instances l with code 23 are instances of a_{23}, and this set of instances would be called L_{23}. Superscript notation is used to denote the turbine the alarm occurred on, so that the set of instances of a_{23} which occurred on turbine 2, is L_{23}^{2}. Each turbine is also assigned a sequential number, and this set of numbers is called T.

4.3.1 Create Batches of Alarm Sequences

The process for creating batches of alarm sequences as they occur during stoppages or periods of curtailment is outlined in algorithm 2.

```
Algorithm 2: Create batches of alarm sequences
Data: Alarms $A$; Alarm instances $L$; Set of turbine numbers $T$
Result: Set of alarm batches $B$
1  Identify all alarm codes which cause the turbine to stop or curtail
     production, $A_r$, and their associated stop categories
2  Identify the alarm code which signifies the turbine returning to normal
     operation, $a_n$
3  for $j$ in $T$ do
4      Find all alarm instances from $L^j$ which have a code in $A_r$. Store set of
         resultant instances as $L_r^j$
5      Find all alarm instances from $L^j$ which have code $a_n$. Store set of
         resultant instances as $L_n^j$
6      Find earliest occurring instance in $L_r^j$. Store its $t_s$ as $t_{b\_start}$
7      Find earliest occurring instance in $L_n^j$ with $t_s > t_{b\_start}$. Store its $t_s$ as
         $t_{b\_end}$
8      Create a batch of alarms, $B_i$, from $L^j$ with $t_s$ that satisfy:

                       $t_{b\_start} \geq t_s < t_{b\_end}$                    (4.2)

9      Find earliest occurring instance in $L_r^j$ with $t_s > t_{b\_end}$. Store its $t_s$ as
         $t_{b\_start}$
10     Repeat steps 7 - 9 until no more instances of $L_r^j$
11 Final step: If two or more batches on the same turbine occur within 1 hour
     of each other, join these "sub-batches" together as one continuous batch
```

The first step is to identify the set of alarms which cause the turbine to stop, or indicate curtailed production, when they appear. This set of alarms is called A_r. From here, each alarm in A_r is assigned a particular "stop category" as follows:

- Fault categories - These are alarms which cause the turbine to shut down due to a fault and are labelled according to the assembly where they originate, corresponding to the Reder taxonomy described in Appendix A. Note that depending on the alarm system, these may differ to the assemblies given in the OEM-assigned categories mentioned in table 4.2. They usually count toward the "Down" availability category seen in table 4.1 or equivalent. Note that the assemblies listed here are non-exhaustive, and only correspond to the assemblies for which there were alarms in the dataset described in section 4.2
    - *yw* - Yaw system faults
    - *ba* - Backup battery system faults
    - *bk* - Blade braking system faults
    - *fc* - Frequency Converter faults
    - *gb* - Gearbox faults
    - *gn* - Generator faults
    - *mi* - Miscellaneous. These are fault alarms from various safety systems that can occur alongside alarms from other systems, e.g. the safety chain
    - *pt* - Pitch system faults
    - *to* - Tower faults. These alarms signify tower structural vibration outside of acceptable bounds.
- *gd* - grid. Alarms that signify the turbine has shut down as the result of a grid issue or fault. Correspond to "Grid" in table 4.1 or equivalent.
- *ma* - manual/maintenance. Alarms that signify the turbine has been manually shut down, e.g. for maintenance or a remote manual shut down. Correspond to "Maint" or "Rep" in table 4.1 or equivalent.
- *no* - normal operation. Alarms that represent times the turbine was shut down or curtailed as part of normal, healthy operation, e.g. curtailed due to grid or noise restrictions, or shut down to perform periodic system tests or shadow-related shut down. Corresponds to "OK" in table 4.1 or equivalent.
- *sn* - sensor. Alarms that signify the turbine has gone down due to a sus-

pected error in one of the turbine's sensors, e.g. wind vane misalignment. Counts as "Down" from table 4.1 or equivalent.

- *wa* - weather. Alarms that signify the turbine control system has shut down or curtailed operation due to severe wind conditions. Note that alarms signifying wind speed below cut-in are not included as in these cases the control system has not intervened to shut down the turbine. Correspond to "Weath" in table 4.1 or equivalent.

Additionally, the alarm which signifies the turbine returning to normal operation is identified. This is usually a single alarm which indicates the turbine is spinning up to generate again, and usually coincides with the turbine returning to the "OK" availability category from table 4.1. This alarm is referred to as a_n. Then, batches of alarms which occur during stoppages on each turbine are created. These include *all* alarms which occur between the turbine stopping/being curtailed, and coming back on-line (as opposed to just A_r alarms). Finally, batches on the same turbine which occur within one hour of each other are joined together as one, continuous batch. This time duration was found heuristically and is done so that if a fault in one sub-system damages another sub-system, with the turbine coming back on-line in between, the two are not treated as separate issues when it comes to labelling the data leading up to the original fault.

A sample of a batch can be seen in table 4.3. Each alarm instance is listed according to its start time, code (note these are randomly assigned), the stop category of the alarm (note that some alarms are not A_r alarms, and so have no stop category), and the associated description.

As can be seen, the batch starts with two frequency converter fault related alarms at around 03:00 am. A number of other alarms are then generated in response to this, including pitch and yaw motor alarms. These "triggered" alarms are not related to the initial fault, but are instead reactions to this initial fault. The maintenance switch is then activated the next day as technicians arrive on site to inspect the turbine. This also causes some additional alarms to trigger. A pitch system test is then performed (often, these periodic tests will automatically be performed when the turbine has been shut down to avoid needing to shut down especially for the test), before the turbine is given the command to start idling and come back on-line. The total duration for this batch was just over 10 hours.

Table 4.3: Example of a batch of alarm sequences

t_s	Code	Stop Cat.	Description
03:02:01	a_1	*fc*	Freq. Conv. voltage fault
	a_2	*fc*	Invalid Response
03:02:40	a_3	*pt*	Blade angle asymmetry
	a_4	*pt*	Pitch Motor Protection
03:03:02	a_5	*pt*	Blade braking time high
13:06:22	a_6	*ma*	Repair Switch
	a_7	*yw*	Yaw motor over temp.
	a_8	*pt*	Pitch Comms Error
13:06:57	a_9	*pt*	Pitch malfunction
13:26:58	a_{10}	*pt*	Pitch System Test
13:33:23	a_{11}	N/A	Idling
13:46:16	a_{12}	N/A	Start-up
13:48:34	a_{13}	N/A	Spinning Up
13:49:57	a_n	N/A	System OK

4.3.2 Assign Stop Categories to Batches

Each batch is then assigned a stop category, similar to how the individual alarms were. This is done primarily by looking at the "root" alarms, i.e. the first alarms that occur simultaneously in the batch (a_1 and a_2 in table 4.3). The rules for assigning stop categories to batches are as follows (items further down the following list supersede those higher up):

- In general, batches are assigned the most common stop category of alarms in the root alarms
- *sn* is assigned if at least one sensor error is present in the root alarms. This is because a sensor error causes the turbine to go down, which in turn can instantaneously trigger other alarms.
- *no* is assigned only if *all* alarms in the root of the batch are *no* alarms; otherwise these alarms are ignored. This is because turbine control systems usually automatically try to schedule system tests during periods of downtime, and these tests do not cause faults themselves. Alarms relating to normal curtailed operation, on the other hand, do not appear with other types of A_r alarms, as the presence of A_r alarms indicates that the turbine has been shut down, as opposed to being curtailed.
- *gd* is assigned if a *gd* alarm appears anywhere in the root. This is because

grid faults can typically cause a cascade of faults to be detected in other systems (even instantaneously from the control system's point of view). This is also assigned if the grid availability counter was active during the stop.

- If the "maintenance" availability counter is active from the time of the root alarms, then the batch is assigned *ma*.
- If the "repair" availability counter is active for any significant duration over the course of the batch, it indicates that some corrective repairs took place. "Significant duration" here can be specified by the user. In this case, the batch is given an additional label, *rep*, as well as the stop category previously assigned.
- Additional rules may be added or edited based on the particular alarms system.

In this way, once the stop categories and rule set have been developed for a particular turbine model, they can be applied to a dataset from a different site or to future generated data from the same site, with no additional manual steps required.

4.4 Results

Batches of alarm sequences were created and labelled using the methodology outlined in sections 4.3.1 and 4.3.2. This resulted in the creation of 1,045 batches, each representing a particular stoppage with an associated turbine number, category, start time, end time, root alarm sequence and list of associated alarm instances. No additional rules for assigning stop categories were needed beyond those in section 4.3.2. Some sample batches can be seen in table 4.4, which shows the start and end time of the stoppage, the turbine number it was associated with and the batch category for that particular batch. The first batch here represents a fault in the yaw system which caused the turbine to go off-line for just under two hours, and no on-site visit was necessary to get the turbine started again. The second batch represents a stoppage that lasted approximately 43 hours, and was due to a fault in the pitch system which needed some on-site intervention to clear.

Maintenance logs were available for all turbine stoppages corresponding to le-

Table 4.4: Sample stoppage batches

Start time	End time	Turbine	Category
2015-11-17 14:59:28	2015-11-17 16:55:57	3	*yw*
2016-04-13 19:51:24	2016-04-15 14:53:11	10	*pt - rep*

vels 3 and 4 of the fault severity categories in section 2.5, and in cases of planned scheduled maintenance activities. The dates and approximate times of site visits were available for each entry, along with a short, unstructured description of the work carried out (e.g. "pitch repairs", "grid works", etc.). In some cases, the actual time of the stoppage was available in the comment, while in others only the time that the maintenance team arrived on site was logged. All entries were assigned a stop category from the description, and where possible matched up with batches. It was found that all maintenance log entries had an associated batch and that the batch categories were correct in every case. For example, for the second batch in table 4.4, the maintenance logs revealed that on the 14th of April a maintenance team arrived on site and found there was a problem with a pitch thyristor, but could not find the source of the fault so left the turbine off overnight until they returned on the 15th of April to fix it. They were off-site by 15:00, corresponding to the time when the batch entry stated the turbine came back on-line.

A single exception was one instance on turbine 8, where the turbine went down due to a pitch fault on 11th January 2016, which was repaired during a routine inspection visit the next day. The maintenance logs simply stated that the turbine was down for a brief routine inspection on 12th January, but the batch creation algorithm correctly identified the stoppage as a pitch fault which resulted in a repair action. This was confirmed as correct by the maintenance team, and is an example of why an automated system can be more reliable and less prone to human error than manually recorded entries.

Further verification of the accuracy of the batches was found through cross-checking with the availability data. All times the "Grid" availability counter was active corresponded to batches labelled *gd*. The same was true of the "Maint" counter and batches labelled *ma*. Importantly, these results mean that because all unplanned maintenance activities (i.e. "severe" faults) and stoppages due to less severe faults were correctly captured by the batch labelling process, the tedious manual step of labelling the data for fault classification can be replaced with this automated process.

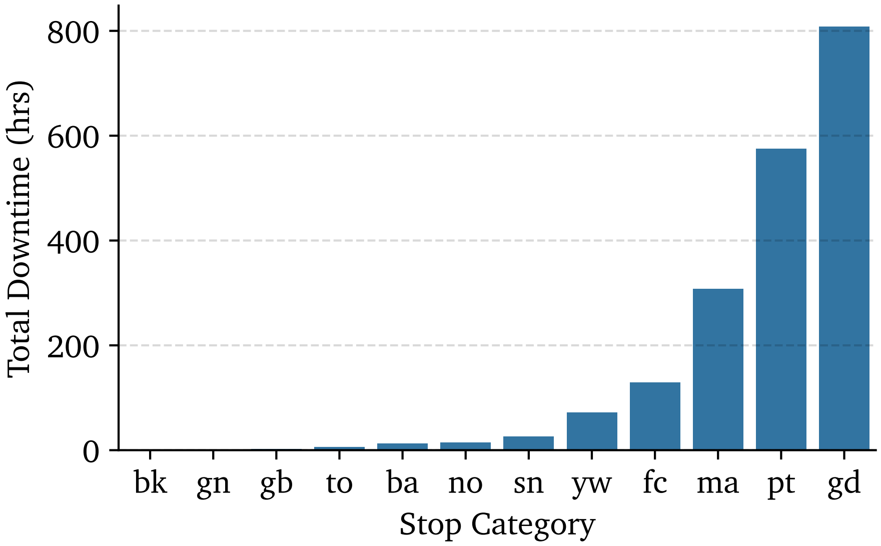


Figure 4.1: Total duration of stoppages for each category of stop

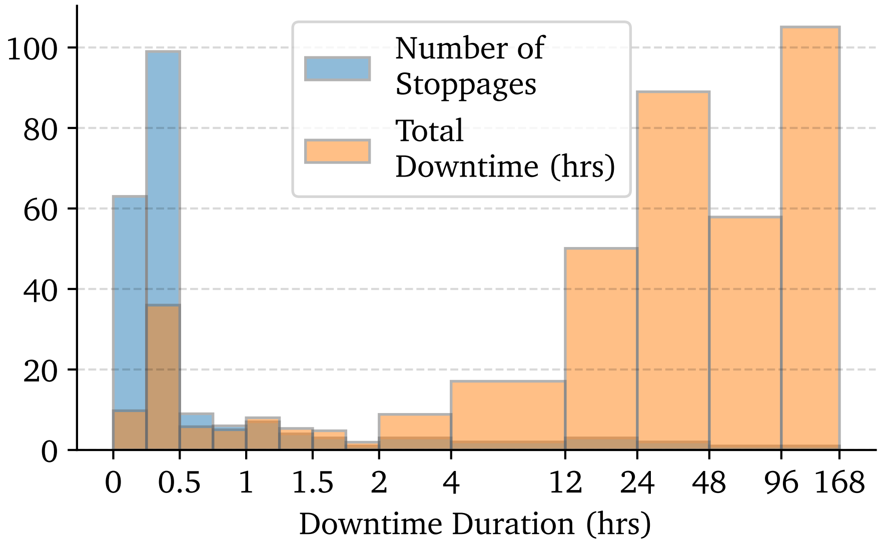


Figure 4.2: Distribution of pitch stoppage durations (note log scale on x axis)

The contribution of each type of stoppage to the total downtime on the turbines is shown in figure 4.1. As can be seen, grid faults were the leading cause of downtime at the wind farm, with pitch faults being the most common cause of turbine-related fault. A histogram showing the distribution of pitch fault durations can be seen in figure 4.2. It is clear that although the vast majority of stoppages are short term in nature, the longer stoppages make up the bulk of the downtime, consistent with findings in reliability studies reviewed in section 2.6. Thus, the utility of such a system in reporting reliability information is demonstrated.

4.5 Conclusions

The work in this chapter presented a methodology for building an accurate database of historical turbine stoppages from SCADA data. Sequences of historical alarms were identified, and rules applied to these sequences to determine the times of and reasons for downtime. In the case of downtime caused by faults, the faults were mapped to the relevant assembly, per the Reder taxonomy described in Appendix A. The methodology was applied to a dataset comprising 6 months' worth of data for 11 turbines, resulting in 1,045 recorded stoppages. The majority of these were very short in nature. Maintenance logs were provided which recorded stoppages which required an on-site intervention to restart the turbine (either a straight manual restart or a repair operation). It was found that the database correctly identified every stoppage that was present in the maintenance logs.

This result means that the resulting stoppage database can be used to accurately label SCADA data for CM purposes. For NBM, stoppages to do with faults, system tests, grid issues, etc. can be removed from the data to build the model. For classification, it means the granularity required for accurate fault labelling can be achieved. The following chapter will incorporate this labelling system as part of a prescribed, modular and robust framework for classification-based CM.

4.5.1 Research Objectives

RO3 of this thesis was:

RO3. Determine whether information on historical failures can be accurately gleaned through analysis of the turbine alarms system, and whether this information can be used to create a complete and accurate training set for fault prediction

This chapter showed that an accurate database of historical failures could indeed be built from historical fault data. The times of failures and the assembly to which they are attributed was found to be accurate by cross-referencing with the fault logs and availability data, making this ideal for labelling the operational data for classification-based fault prediction. Hence, RO3 has been achieved.

Chapter 5

Robust Framework and Evaluation Criteria for Classification-Based CM

5.1 Introduction

In chapter 3, a case study was performed on classification using single instances of specific alarms as references for when faults occurred. However, this gives a less accurate picture of the type and severity of the faults compared to the more robust methodology developed in chapter 4. This provided more clarity and granularity as to what assembly the faults occurred in, how long they lasted and whether a site-visit was required. Furthermore, although classification scores were given in chapter 3, these scores do not provide a holistic picture of how well the solution would perform as a field-deployed system; if a single live point is classified as looking like a fault may be imminent, but the next subsequent points are not, should a maintenance investigation take place? Using a sliding window metric, whereby the portion of samples predicted as unhealthy in a certain time window is measured, can help smooth out any noise and provide operators with more robust decision support (Zhao et al. 2017).

In this chapter, a formal framework for CM using classification approaches based on SCADA data will be presented, leveraging some of the techniques and findings developed in chapters 3 and 4. The framework will incorporate the labelling process outlined in chapter 4, and introduce a window-based alarm system in order to assess the performance of any models produced by the framework in a real-world setting. The developed framework will be prescriptive and robust (i.e. incorporating machine learning best practices), while still being modular enough to fit the needs of future researchers in the space. It aims to avoid some of the common pitfalls seen in classification-based approaches in the literature (as described in section 2.8.4), as well as provide a way to assess real-world performance for fault prognosis. Unlike the work in chapter 3, this framework will focus exclusively on the fault prognosis aspect of CM, as opposed to fault detection or diagnosis (chapter 6 will investigate ways of more accurately diagnosing faults using alarm system data). The framework will be applied and demonstrated on the same set of data used in chapter 4.

As mentioned previously in this thesis, what constitutes a fault has different definitions in different parts of the literature. In the review of reliability studies in section 2.6, faults were variously defined as stoppages that lasted more than an hour, stoppages where a maintenance call-out was required to restart the turbine, or stoppages where material was consumed in order to get the turbine running again (i.e. repairs or replacements took place). The methodology in chapter 4 provided information related to the duration of stoppages and also information on whether or not repairs took place. Because of this, the work in this chapter will evaluate the fault prognostics framework using different criteria for what constitutes a failure. This will be based on the length of the stoppage or whether or not a site-visit was required to restart the turbine. Included in these site-visits are straightforward manual restarts which fixed the issue - however, as mentioned in section 2.7.3, in the offshore context such manual restarts can be a lot less straightforward due to access issues. Hence, no differentiation will be made between faults where actual repairs took place, and where a manual restart was required.

5.2 Framework

The framework developed in this work can be split into three main parts, seen in figure 5.1: (i) alarm sequence batch creation; (ii) data labelling & classification; and (iii) fault prediction system deployment. Part (i) deals with identifying "batches" of alarm sequences as they appear during turbine stoppages, and assigning a particular "stop category" to give a reason for each stoppage. This represents the work done in chapter 4. Part (ii) deals with using these batches to label SCADA data and perform classification on this data. Part (iii) deals with deploying the trained classifier in the field and using a "sliding window" metric for notifying operators of possible impending faults.

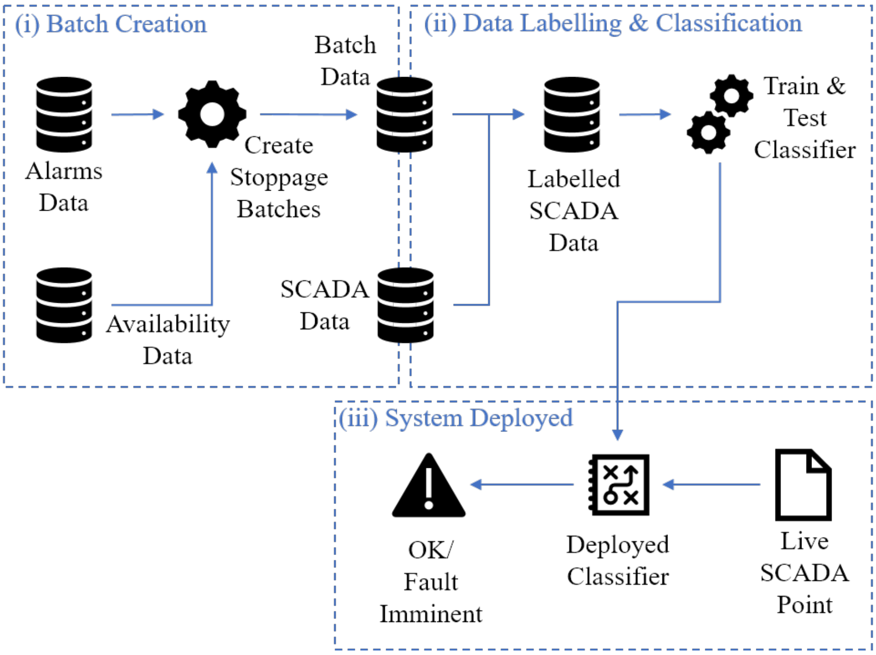


Figure 5.1: Overview of Overall Framework

5.2.1 Alarm Sequence Batch Creation

The alarm sequence batch creation involves building a database of historical stoppage data. This includes the start and end time of every planned or unplanned stoppage that each turbine experienced, as well as giving a reason for the stoppage. This step represents the methodology covered in chapter 4.

5.2.2 Data Labelling & Classification

The general methodology for this part of the framework is shown in figure 5.2. As can be seen, this stage can be separated into two broad steps; labelling, and the classification itself.

5.2.2.1 Data Labelling

In this framework, the classification being attempted is to differentiate between *pre-fault* and *healthy* data. That is, trying to distinguish between data leading up to a fault (when it can be assumed that this fault was developing), and otherwise healthy data.

Data Cleaning: Note that prior to any work being done, the SCADA data itself

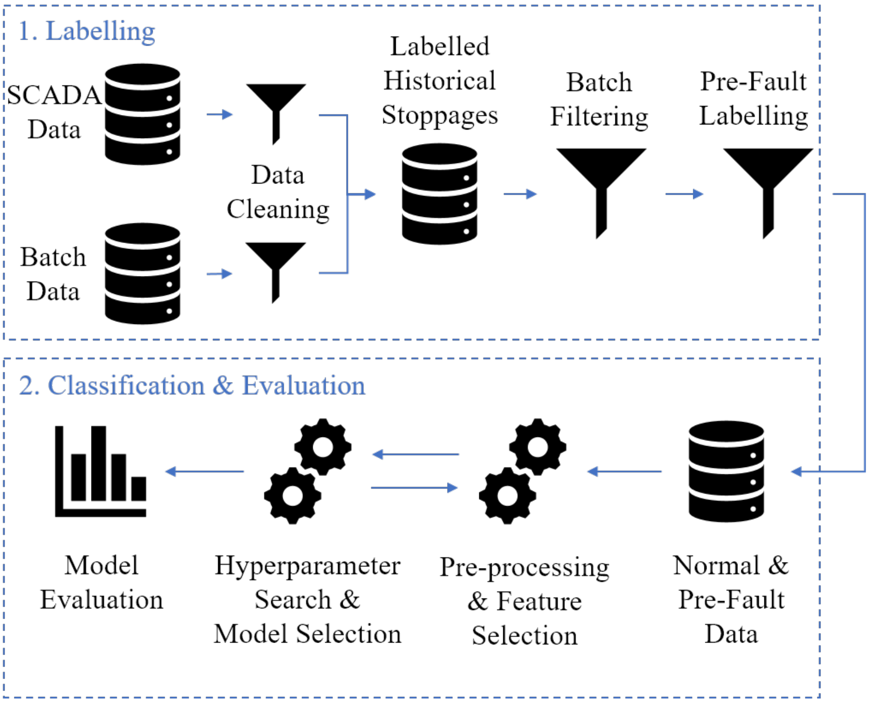

1. Labelling
SCADA Data
Batch Data
Data Cleaning
Labelled Historical Stoppages
Batch Filtering
Pre-Fault Labelling
2. Classification & Evaluation
Model Evaluation
Hyperparameter Search & Model Selection
Pre-processing & Feature Selection
Normal & Pre-Fault Data

Figure 5.2: Overview of step "Data Labelling and Classification"

must be cleaned. Any variables that have very little variance, e.g. as a result of frozen sensors, are removed. As well as this, any variables that look to have a significant proportion of erroneous or missing values are also removed, e.g. if 10% of the nacelle ambient temperature values are missing or implausible, this should be removed. Finally, individual samples with missing or erroneous values should be removed. For example, if there are a small number of samples with implausible power output values, these should be removed. The specific criteria for removing values will vary from dataset to dataset.

Label Historical Stoppages: Next, the batch stoppage data (obtained in step 5.2.1) is overlaid onto the SCADA data to label each 10-minute data point as being "normal/healthy" or as being associated with a particular batch stop category (such as those mentioned in section 4.3.2), e.g. *gd*, *pt*, *no*, etc.

Batch Filtering: A filtering step is then applied whereby SCADA data relating to certain types of batches are filtered out so that only healthy data and the particular type(s) of faults to be detected are left. *wa* stoppages and curtailments should be left included, as the resulting classifier should be able to take into account adverse weather conditions to avoid throwing false positives. Of the remaining fault data, stoppages above or below a certain duration may also be excluded for the purposes of trying to predict more or less severe faults. In a similar vein, only stoppages where corrective repairs took place may be included. For example, if trying to predict pitch faults which resulted in corrective repairs, all stoppages except those labelled both *pt* and *rep* are removed from the data. This differs from the classification in chapter 3, where certain types of faults were detected against a backdrop of a wide array of operational data, including other faults. This addition to the framework was made due to better performance seen in early results when filtering out other types of stoppages. This would not be possible without the more granular stoppage data obtained in step 5.2.1.

Pre-Fault Labelling: Finally, SCADA samples preceding each remaining fault-stoppage at time t are given label y:

$$y_t = \begin{cases} \text{F}, & \text{if } t_s - w_2 - 600s < t \leq t_e + 600s \\ \text{PF}, & \text{if } t_s - 600s - w_1 < t \leq t_s - 600s - w_2 \\ \text{NF}, & \text{otherwise} \end{cases} \tag{5.1}$$

where $\{w_1, w_2; w_1 > w_2\} \in \mathbb{N}^+$ are windows of time in advance of the stoppage, t_s is the start time of the batch, t_e is the end time of the batch, F is the label of the stoppage itself, PF is the pre-fault data, and NF represents healthy, fault free data. In this way, a window of time w_1 preceding the fault is labelled as pre-fault data, while leaving a window w_2 as a "minimum prediction time" before the fault occurs. The time between $t_s - w_1$ and $t_s - w_2$ is known as the "pre-fault window". Note that a 600s time band is once again added to ensure faults which only lasted part of a sample are labelled as such. Finally, all data labelled F is removed, to only be left with NF and PF data to train the classifier.

5.2.2.2 Classification

The classification step follows a typical machine learning process. This process can be quite heuristic and be dataset-dependent, so only a high-level overview will be presented. In any case, current machine learning best practices should be employed (James et al. 2013).

A number of classification algorithms can be used, such as Support Vector Machines (SVMs), Random Forests (RFs), Logistic Regression (LR), Artificial Neural Networks (ANNs), etc. Depending on the classification algorithm selected, the data may need to be normalised. As well as this, over or undersampling methods may be used to reduce the imbalance in the dataset. Typical feature selection/extraction or dimensionality reduction methods, such as polynomial feature generation or principal component analysis (PCA), can be used to identify important features. Domain knowledge can also be applied to create relevant features (e.g. differences between certain parameters, averages, time-lagged features, etc.). A case-study of feature engineering for this domain can be found in (Hu et al. 2016).

Next, cross-validation is employed to search over model hyperparameters and compare scores across different algorithms. Finally, the best performing model is tested on a held-out test set or another appropriate cross-validation scheme, using appropriate scoring metrics. As will be seen in section 5.3, the classification process can be repeated any number of times with different batch filtering techniques in order to evaluate effective prediction windows and which types of faults can be predicted.

As mentioned earlier, there are many approaches that can be taken in this iterative process to achieve the best results, so no prescribed set of steps is presented at this stage of the overall framework. However, as discussed in section 2.8.4, there are some pitfalls seen in some parts of the literature that should be avoided. Some of these are presented here.

Cross Validation Scheme: The experimental layout used for obtaining cross-validation scores across models, as well as the final score, must be appropriate. Cawley and Talbot present some of the best practices and common mistakes to take note of when employing cross validation (Cawley & Talbot 2010). For example, nested cross-validation or a separate test set should be used to avoid using the same data to tune hyperparameters and evaluate the model, which can lead to inflated scores.

An appropriate scheme to deal with the fact that turbine time-series data are not independently and identically distributed should be selected. Using a typical shuffled K-Fold cross validation scheme will lead to greatly inflated scores due to the strong auto-correlation of samples that are close in time to each other. One way of addressing this is by using a modified version of k-fold which returns the first k folds as a training set, and the $(k+1)$th fold as the test set. The difference here is that successive training sets are supersets of those that came before.

Another way of doing this, as described by Arlot and Celisse in (Arlot & Celisse 2010) is to modify k-fold to choose training and validation folds $I^{(t)}$ and $I^{(v)}$ such that:

$$\min_{i \in I^{(t)}, j \in I^{(v)}} |i - j| > h > 0 \tag{5.2}$$

where h is some minimum distance between samples such that samples x_i and x_j are independent.

Something to keep in mind if using either of these modifications of k-fold is to make sure that no split occurs across PF data relating to a specific fault instance, i.e. that some samples of PF data from the same fault end up in both the training and validation set.

An alternative scheme not based on k-fold, and one which is seen more widely in literature, would be to use some set of turbines for training, and another set for validation. If this is the case, the final model should be tested and averaged across a number of different turbines to get a better picture of how the model generalises to unseen data.

Undersampling & Transformations: The validation or test set should never be under/over sampled in any way, or any information about it used in the sampling process. The true distribution of the data must remain in the final scoring stage. Similarly, if the data is being normalised, this should be done on the training set, with the transformation parameters stored and applied to the validation set, so that absolutely no information about the test set can leak into the training set.

Scoring Metrics: The scoring metrics on the final model should incorporate a way of measuring both the false positive and false negative rate; i.e. false alarms and missed faults. Precision and recall are the recommended ways of achieving this.

5.2.3 Fault Prediction System Deployment

Once the classification model has been built and deployed, new live data points are fed to it every 10 minutes. The classifier then decides whether these points are PF or NF. The final part of this framework centres on using this information to create a metric for alerting maintenance technicians of impending faults, and simulating the real world accuracy of this system as it would be deployed in the field.

The proposed metric for generating actionable fault prediction alerts is as follows:

$$M(t) = \sum_{i=t-w_m}^{t} y_i \tag{5.3}$$

$$\text{where } y_i = \begin{cases} 1, \text{if label is PF.} \\ 0, \text{if label is NF.} \end{cases}$$

where $M(t)$ is the value of the alert metric at time t, y_i is the label given by the classifier for samples at time stamps i, and w_m is a window of time of predetermined length. An "impending fault" alert is triggered whenever $M(t)$ is greater than some threshold b. In this way, the system is made more robust to false positives due to needing a number of positive results being triggered within a certain window of time.

This system can be accurately tested by performing classification on a held-out test set (or using nested cross validation), feeding the resulting labels one-by-one to the alarm function, and checking how many times false alarms were raised or faults were correctly identified in advance. This will then give a confidence score for how well the system is expected to perform while deployed.

5.3 Application: Case Study & Results

This section will be separated into three parts, mirroring each step of the process outlined in figure 5.1. The results of applying the methodology to the dataset outlined in section 4.2 will be presented, and the implications of these results will be discussed.

5.3.1 Create Batches of Alarm Sequences

The first step of the methodology represents the work presented in chapter 4. The results of this are detailed in section 4.4. In summary, there were 1,045 batches in the data, each representing a particular type of stoppage with an associated duration and, if the stoppage was fault-related, whether or not ma-

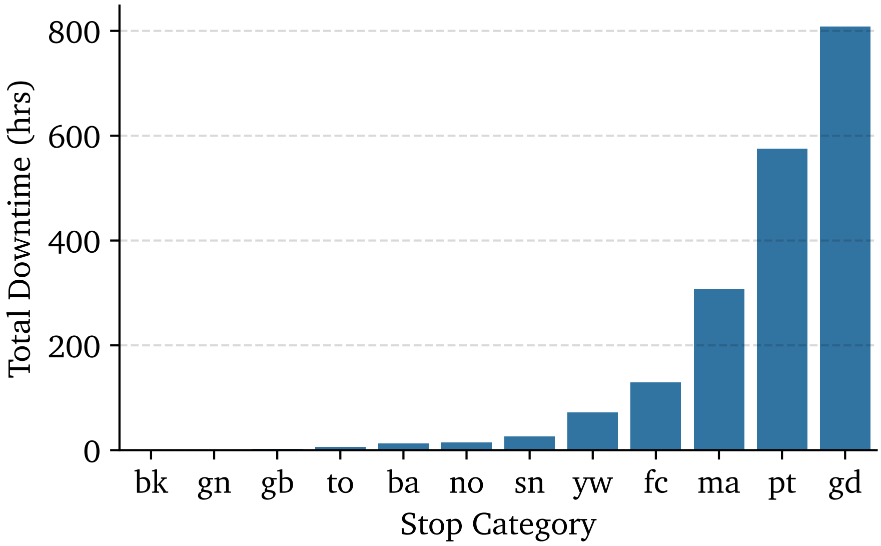


Figure 5.3: Total duration of stoppages for each category of stop

nual intervention was needed in order to restart the turbine. This database was found to be accurate by cross-referencing with availability data and fault logs.

Figure 5.3 shows the duration of stoppages for each stoppage category. As can be seen, grid-related faults had the biggest contribution to downtime. For component-related faults, the pitch system was the biggest contributor, followed by the frequency converter and yaw system. It was found that the frequency converter and yaw system faults were related to a small number of high duration faults, with too few samples for effective classification to be performed. For this reason, only pitch system faults were focused on.

5.3.2 Data Labelling & Classification

Next, the alarm batches representing stoppages were overlaid on the SCADA data in order to predict when pitch faults are to occur with some advance notice. The SCADA data was first cleaned by removing columns (i.e. features) with missing, frozen or implausible values (e.g. temperature sensors consistently reading below absolute zero). Next, rows (i.e. samples) with missing values were removed.

As mentioned in section 5.2.2.1, all stoppages except those corresponding to the faults being predicted (in this case *pt* batches) were removed from the data. The fault data contained faults with a wide range of durations, and included faults which resulted in both manual intervention and remote restarts. For this reason, a number of separate cases were investigated, which further filtered out the remaining *pt* depending on certain criteria. In total, there were 7 different cases, where stoppages which did not meet the following criteria were filtered out:

- $>$ 0 hours duration
- $>$ 1 hours duration
- $>$ 2 hours duration
- $>$ 6 hours duration
- $>$ 12 hours duration
- $>$ 24 hours duration
- stoppages where repairs were needed

Each of these cases were tried with 6 different pre-fault windows (i.e. w_1 and w_2 from equation 5.1), for a total of 42 different classifiers. These windows are listed below (all durations in hours):

- $w_1 = 2$, $w_2 = 0$
- $w_1 = 4$, $w_2 = 0$
- $w_1 = 8$, $w_2 = 0$
- $w_1 = 24$, $w_2 = 0$
- $w_1 = 24$, $w_2 = 6$
- $w_1 = 48$, $w_2 = 6$

The set of features from the SCADA data for the classification stage was based on the features selected in (Kusiak & Verma 2011, Chen et al. 2011, Godwin & Matthews 2013), with some additions. This amounted to the following features:

- Wind speed (average, max. & standard deviation)
- Average blade angle for each blade
- Set torque
- Real Power Output

Additionally, the following computed features were used:

- The absolute difference between each pair of actual blade angles (e.g. blade 1 and blade 2)

The absolute differences were computed by converting angles to their sine and cosine components. One turbine was randomly selected as the test set, with the other ten being used as the training/validation set. Importantly, the test set was held out completely until simulating alarm system deployment. A number of different classification algorithms were initially compared as a first pass evaluation on a subset of the training data, including support vector machines (with linear, polynomial and Gaussian kernels), decision trees and logistic regression. It was found that the support vector machines with Gaussian kernel and decision tree classifiers performed best. Both had similar scores, but the decision trees were much quicker to train and also gave human-readable rules. Ensembles of decision trees were found to improve performance, so a random forest classifier was the final type of classifier used to train all models. Details of this algorithm can be found in (Breiman 2001).

After this, a hyperparameter search was performed for the case of $w_1 = 48$ hours and $w_2 = 6$ hours, with no minimum stoppage duration (i.e. all batches were included). This minimum batch duration and pre-fault window were chosen so as to get the maximum amount of fault samples in the training data, while getting a good advance warning of imminent faults. The search was performed over the ten turbines in the training set by splitting the data into train/validation folds according to turbine number, with a single turbine in the validation fold each time.

It was found that randomly undersampling the training data and using 2D principal component analysis performed best, with the number of trees in the random forest nodes set to 5 and the maximum number of features at each split set to 11. With the optimal hyperparameters found, the same cross validation technique was used to compare scores across all other cases of window length/minimum batch duration. The results of this are shown in figure 5.4, with the scores representing the average precision and recall across all folds.

As can be seen, the precision and recall scores are generally quite poor. The general trend shows that best scores are seen when all batches regardless of stoppage duration are included, and with a longer pre-fault window. A possible reason for this may be that there are simply more training examples of the PF class for these labelling scenarios. The best scores were a precision of 0.155 and recall of 0.49, and this was seen in the case where there was no minimum batch

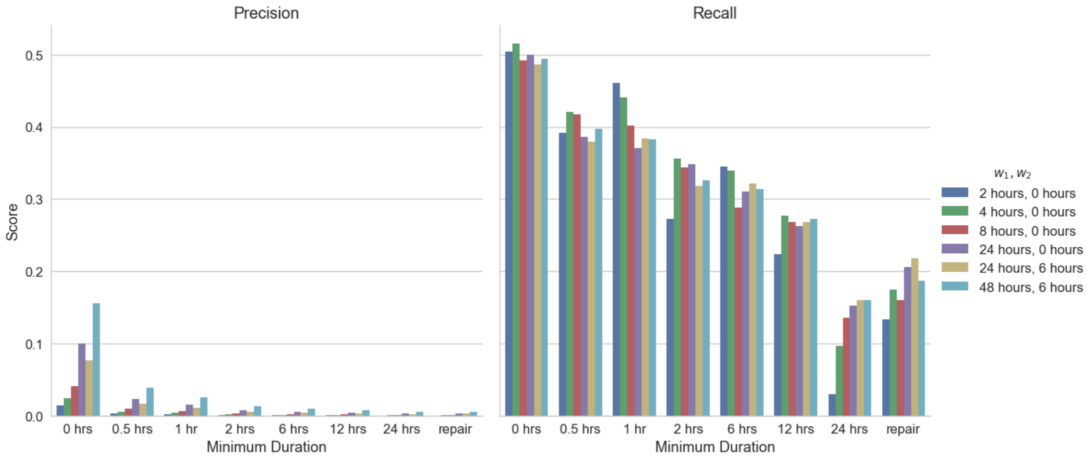


Figure 5.4: Cross-validation results for various minimum durations of downtime and pre-fault windows w_1 and w_2

duration, and pre-fault windows of $w_1 = 48$ hours and $w_2 = 6$ hours. Intuitively, it would be expected that faults where repairs are needed would have a strong leading fault signature due to physical degradation being presumably present leading up to the fault. However, this was not the case almost surely because of the very few training examples of pitch faults which resulted in repairs (only 7 such instances) in the data.

As discussed in section 2.8.4, recall scores in the literature of up to 87% were seen 48 hours in advance of a fault, with precision scores of 17% (Godwin & Matthews 2013). The reasons for the comparatively poorer scores seen here are likely due to a smaller amount of data being available, or due to the faults on the models of turbine used in this dataset being inherently harder to detect.

5.3.3 System Deployment

In order to simulate the fault alert system being used in real world deployment, two of the above cases were selected to re-train the optimal model using the full set of training data and evaluating on the held-out test set. The alarm system's effectiveness with the following pre-fault windows and minimum batch durations was investigated:

- $w_1 = 48$ hours, $w_2 = 6$ hours with no minimum batch duration, and
- $w_1 = 48$ hours, $w_2 = 6$ hours and minimum batch durations of 30 minutes
- $w_1 = 48$ hours, $w_2 = 6$ hours and minimum batch durations of 1 hour

Table 5.1: Results of alarm system evaluation (lighter colours represent better scores on that column)

min. dur. (hrs)	stops	prec.	rec.	w_m	b	avg. notice (hrs)	pred. (%)	# false	false dur. (hrs)
0	66	.46	.51	12	12	28	9	114	19
0	66	.46	.51	12	10	38	64	745	124
0	66	.46	.51	144	108	46	3	187	31
0	66	.46	.51	144	96	34	5	452	75
.50	13	.12	.50	12	12	34	8	16	3
.50	13	.12	.50	12	10	31	62	714	119
.50	13	.12	.50	144	108	0	0	0	0
.50	13	.12	.50	144	96	21	8	10	2
1	7	.06	.47	12	12	0	0	11	2
1	7	.06	.47	12	10	35	71	729	122

The first case was selected as this is what achieved the best classification results, and the second and third cases were selected in order to try and see if stoppages which cause lengthy downtime can be avoided. Values of w_m of 12 and 144 time steps were used, corresponding to 2 and 24 hours (due to each time step lasting 10 minutes). The threshold b was varied also. This time, the full set of training data was used and the held out test set (randomly selected as turbine 2), was used for evaluation.

The results are shown in table 5.1. Here, *stops* refers to the number of stoppages which were present in the test set for that particular case. *prec.* and *rec.* refer to the precision and recall scores of individual samples achieved in the classification stage, and lighter cells in these columns represent better scores. *avg. notice* refers to the average amount of warning, in hours, that was given from when an alert was raised to when the stoppage occurred. *pred. (%)* refers to the % of the stoppages in the test set which were successfully predicted. *false (#)* refers to the number of false alerts which were raised, while *false dur. (hours)* refers to the total duration that the alarm was active over the full year of data where no stoppage was imminent. For the last four columns, lighter colours once again indicate better scores on that column.

As can be seen, $w_m = 12$ produced far better results than $w_m = 144$ in all cases, with a higher % of predicted stoppages and a lower rate of false alerts. The precision score for a minimum duration of 0 hours was drastically higher in the test set than in the cross-validation stage, at .46 - an improvement on scores seen in literature. Surprisingly, with $w_m = 12$ and $b = 10$, the % of predicted faults and false alarm rates were similar for the cases of no minimum batch duration, a minimum duration of 30 minutes, and a minimum of 60 minutes, despite the significantly worse classification scores in the latter cases. The % of predicted batches ranged from 62% to 71% with an average notice of between 31 and 38 hours given, and the duration of false alarms was between 119 and 124 hours. In both the case of no minimum duration and a 30 minute minimum, setting $b = 12$ lowered the duration false alarms to as little as 2 ours, but expectedly reduced the number of predicted faults to 8%.

5.4 Conclusions

This work presents a framework for building and deploying a fault prediction system using wind turbine SCADA data in three parts, and evaluates the framework with a case study. The first part describes a novel method to build an accurate database of training data by automatically identifying sequences of historical turbine alarms, and using a rule set to infer times of and reasons for downtime on the turbine. The details of this step were covered in chapter 4. Next, an overview of how to label and filter data and apply classification techniques relevant to this domain was given, and common pitfalls that are seen in literature which can lead to inflated test set or cross-validation scores were discussed. Finally, a novel sliding window metric for alerting maintenance technicians of impending faults was proposed, with tunable parameters which allow tweaking of caught faults vs. false alarms.

The framework was applied to a dataset comprising 6 months' of data for 11 turbines. When the classification methods were applied, cross validation scores were generally quite poor - in the range of .15 precision and .49 recall. However, these increased to .46 and .51, respectively, when trained on the full set of data and tested on the held-out test set. The sliding window metric performed very well and was able to detect up to 71% of faults on the test turbine 35 hours' in advance. The alarm was active for 122 hours of the year when no fault was imminent. Adjusting the threshold for creating a fault alert reduced the time the alert was erroneously triggered to 2 hours but only predicted 8%

of faults. These results show that faults can be predicted in advance, though a trade-off has to be made between the number of successfully predicted faults and false alarms.

The value of a system such as this becomes clear when severe faults (i.e. faults that needed a repair action or caused significant down time) can be predicted. Unfortunately, the low volume of data used in this study meant that there were few samples of severe fault events for the classifiers to be effectively trained, and so the alarm system did not perform well on trying to exclusively predict these types of faults. However, the results show that this system works in principle by predicting less severe faults, even when raw classification scores are not very high. Furthermore, more training data will improve both precision and recall scores in the classification stage, and hence allow the alarm system to perform more effectively.

As discussed in the text, although the classification scores do not approach some of the scores found in some of the literature, some of these scores are inflated due to flaws in the machine learning experimental set-up. This framework has been designed to be modular and scores are very much dataset-dependent, so it is hoped other researchers can apply or modify it in their own work to see if classification scores can be improved with bigger datasets, while avoiding some of the commonly seen pitfalls in the literature.

Having more fault examples should drastically increase fault prediction scores, so future work will focus on applying the framework to a bigger dataset. This would allow the incorporation of RUL estimates and a confidence score for the alarms generated in the sliding window metric. Finally, since such a system requires little to no capital expenditure, the value of reducing the number of predicted faults vs. the expenses incurred in false maintenance call-outs should be explored to find an economic optimum.

5.4.1 Research Objectives

ROs 1 and 4 of this thesis were as follows:

RO1. Determine what level of condition monitoring can be performed using classification techniques

RO4. Design a comprehensive framework which incorporates all previous findings as well as best practices from literature and apply this methodology to evaluate its performance as a field-deployed system

This chapter showed that condition monitoring at a prognostic level was possible using classification techniques - advance notice of impending pitch faults was given. The system could be tuned to reduce the number of false alerts so that any alarms generated were done with high confidence (at the expense of missing some faults). Hence, RO1 was achieved.

The CM was achieved by developing a prescribed framework, complete with guidelines to best practices, and applying it to an existing dataset. A held-out test set was used to simulate real-world performance using a sliding window metric to generate impending fault alerts, and this performance was evaluated. Hence, RO4 was also achieved.

Chapter 6

Additional Diagnostic Functionality from the Alarm System

6.1 Introduction

In previous chapters, methodologies were presented which focused on fault prognostics to prevent unplanned stoppages. However, even with perfect CM, some amount of unplanned stoppages are inevitable. In this chapter, the focus will be on providing tools to operators to be able to glean more information from the turbine SCADA alarms system about faults which have occurred, in order to help with diagnosis of those faults and plan an appropriate maintenance action.

In (Qiu et al. 2012), the authors performed a detailed study on turbine alarm systems, applying standards used in the oil and gas industry (Noyes 1999). They identified 3 KPIs for alarm generation:

- The average number of alarms occurring per 10-minute SCADA period
- The maximum number of alarms which occurred in any 10-minute SCADA period
- A histogram of alarm frequency per 10-minute period, split into a number of different bins (0, 1-10, 11-50, >51)

They applied these KPIs to four different wind farms, and found that in all four the KPIs exceed acceptable bounds. In three of the wind farms, the alarms were found to be unmanageable during a fault event, when high volumes of alarms known as "alarm showers" are generated. For the fourth wind farm, it was found that the alarms were not only not useful, but also can present an unhelpful distraction over a longer period. High alarm rates such as these not only prevent any useful information from being presented to the user, but can also reduce operator sensitivity to malfunctions, leading to potentially catastrophic consequences (Noyes 1999). The authors in (Chen et al. 2011, Gonzalez et al. 2016) corroborate these findings and present some solutions to the alarm showers generated during fault events, discussed in section 2.8.3.

One of the issues with high volumes of generated alarms means that it can be difficult to to know which alarms are related to the initial fault, and which alarms are *reactions* to other alarms generated as the fault propagates through the turbine (Gonzalez et al. 2016). In the case of faults in the pitch system, for example, if a pitch motor fault is detected, there are several contingency measures in place that kick in to avoid emergency situations. These include emergency brakes to stop the blades turning in case of a storm, and backup batteries in case power supply to the turbine is interrupted. Hence, if a pitch motor fault occurs, a number of alarms are generated to give information about the status of the auxiliary systems, or even faults in these auxiliary systems themselves, along with alarms related to the original fault. Without access to quality documentation and an intimate knowledge of the particular alarm system, it can, in some cases, be hard to decide which alarms are attributable to the original pitch motor fault, and which are to do with the auxiliary systems. Hence, an expert system which can decode some of this knowledge would be useful for operators.

As discussed in section 2.6, less severe faults which result in short stoppages can occur quite frequently on turbines. However, if certain types of alarms or alarm sequences are occurring frequently they can be indicative of a more severe issue with the turbine than the individual stoppages may imply. Because these are often often cleared with an automatic reset, they can be overlooked by operators who may feel the effort required to analyse the density of alarms generated in such events is outweighed by the actual impact on turbine availability that any single short stoppage incurs.

In this work, a similar approach to that of chapter 4 is taken to identify batches of alarm sequences related to a particular turbine assembly as they occur during stoppages. Clustering techniques are then applied to group certain batches together which contain similar alarm sequences. In this way, each stoppage can be attributed to a specific type of sequence with its associated characteristics, rather than a large number of individual alarms which must be analysed. This reduces the burden of analysis for technicians as stoppages related to specific alarm sequences can be investigated for shared characteristics. This would allow information such as the probable root cause of the stoppage to be instantly known whenever such an alarm sequence reappears in future.

6.2 Description of Data

The data used in this study comes from an Irish wind farm with eleven 2.5 MW Doubly Fed Induction Generator (DFIG) turbines. The study covers a period of eleven months from June 2015 to April 2016. There were 118 days across all turbines where a maintenance team was on-site during this period. 56 of these days were due to 35 individual fault instances on the turbines which could not be fixed or diagnosed remotely. The remaining 62 days were due to scheduled periodic maintenance or upgrade work. Stoppages which did not require a maintenance call-out, e.g. when the turbine went down due to a fault which could be corrected remotely, were not recorded by the operator. The data used was alarm data from the turbines' OEM alarm system. The turbines were the same model as those described in section 4.2, and so used the same alarm system as described there.

6.3 Methodology

The methodology developed in this research is split into two broad parts and summarised in figure 6.1. The first part focuses on identifying alarms relevant to potential faults which could occur in a particular assembly (e.g. the pitch system or frequency converter). A single assembly is focused on at a time in order to reduce the complexity of the analysis.

The second part of the methodology then focuses on identifying sequences, or "batches", of these alarms as they appeared during stoppages related to faults in the assembly, and using cluster analysis to group similar batches together. In this way, stoppages which share a similar sequence of relevant alarms can be grouped together. Further investigation can then be performed in order to identify a root cause for these stoppages. When a particular alarm sequence appears in future, the root cause will be known with minimal further analysis needed. The rest of this section is split into detailed sub-sections which correspond to each of the steps outlined in figure 6.1.

The terminology and notation used in this chapter is similar to that described in chapter 4. Hence, "alarm instance" refers to an individual alarm in the dataset, not to be confused with "alarm", which refers to that type of alarm (as opposed to a specific instance of it), or "alarm code", which refers to the code for that alarm. The full set of alarms is labelled A, with individual alarms labelled a_c, where c represents the code for that alarm. Alarm codes, and in some cases descriptions, have been changed for purposes of anonymity, and are referred to as a_1, a_2, a_3, etc. Similarly, L refers to the full set of alarm instances, l refers to individual instances. t_s refers to the start time of an alarm, and t_e the end time.

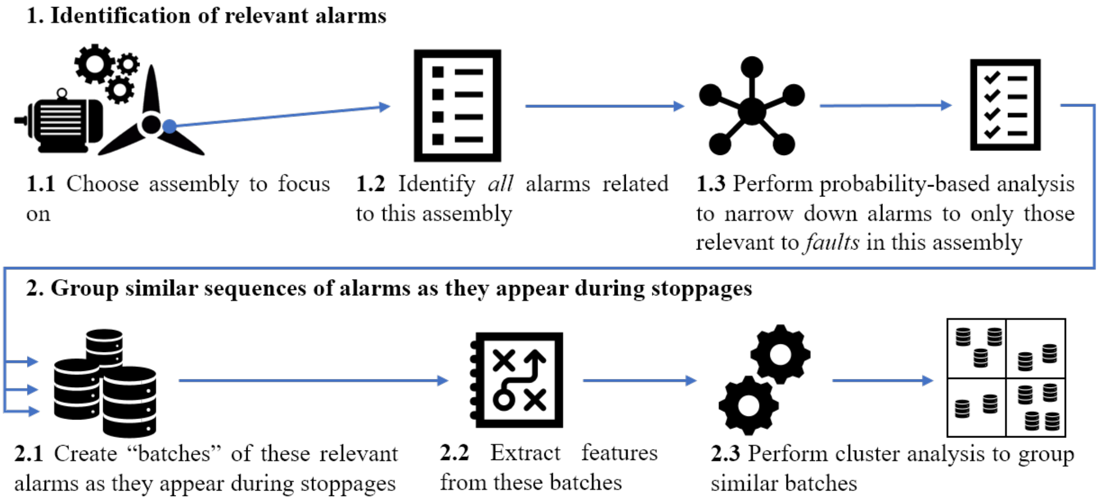

1. Identification of relevant alarms
1.1 Choose assembly to focus on
1.2 Identify *all* alarms related to this assembly
1.3 Perform probability-based analysis to narrow down alarms to only those relevant to *faults* in this assembly
2. Group similar sequences of alarms as they appear during stoppages
2.1 Create "batches" of these relevant alarms as they appear during stoppages
2.2 Extract features from these batches
2.3 Perform cluster analysis to group similar batches

Figure 6.1: Methodology overview

6.3.1 Identification of Relevant Alarms

6.3.1.1 Choose assembly to focus on

In order to reduce the complexity for analysis, a single turbine assembly at a time is analysed in this methodology, e.g. frequency converter, generator, pitch system, etc. Figure 6.2 shows the frequency of fault alarms according to their OEM-assigned category in the alarm system. These alarms are from all eleven turbines over the eleven-month period of the study, as detailed in section 6.2. As can be seen, the pitch system has the most frequent fault alarms. For this reason, this assembly is focused on in this study.

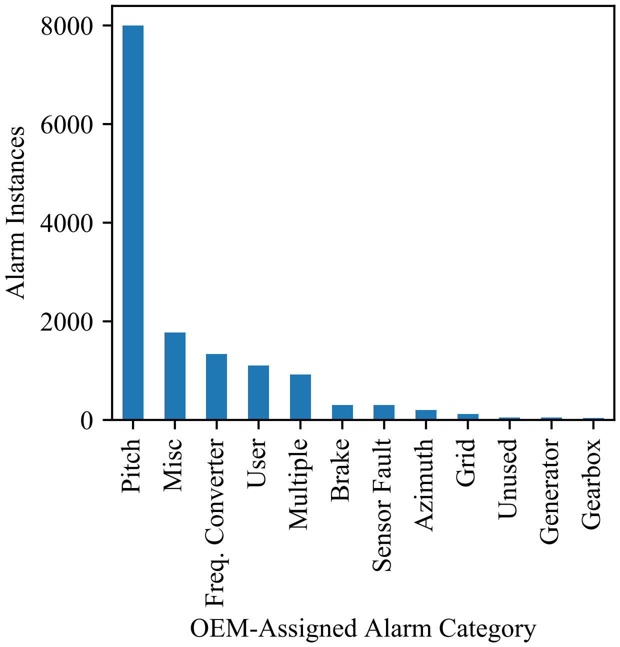


Figure 6.2: Number of alarm instances with type "Fault" or "Critical Fault" by OEM-assigned category

6.3.1.2 Identify *all* alarms related to this assembly

As stated in the introduction, identifying alarms related to a specific assembly can be complicated by the volume of alarms and, in some cases, lack of clarity on their function or the relevant part of the taxonomy to which they belong. Hence, a probability-based analysis as described in (Qiu et al. 2012) and (Gonzalez et al. 2016) is performed which gives insights into groups of related alarms and which alarms trigger each other. This gives an easy to interpret visual aid for what alarms could be important in determining periods of faulty operation related to a particular assembly.

Before doing the probability analysis, *all* possible alarms related to the assembly in question are identified. These include information, warning and fault (including critical fault) alarms related to the assembly itself as well as the auxiliary and support systems. If there are certain alarms where it is not clear to which system they belong to, they should be included anyway.

In this case, all alarms relating to the pitch system and its auxiliary support systems were included, for a total of 58 alarms, referred to here as A.

6.3.1.3 Perform probability-based analysis to narrow down alarms to only those relevant to faults in this assembly

The probability based analysis is performed as follows:

1. From the set of alarms A, all combinations of pairs of alarm codes are found, $\binom{A}{2}$.

2. For each pair of alarm codes a_1 and a_2, count the number of instances of alarm a_1 which have triggered one or more instances of a_2 and vice-versa.

   An instance of a_1 is said to trigger an instance of a_2 if the following conditions are met:

   $$t_{s_{a1}} <= t_{s_{a2}} \wedge t_{e_{a1}} >= t_{s_{a2}} \tag{6.1}$$

   where $t_{s_{a1}}$, $t_{s_{a2}}$ and $t_{e_{a1}}$ are the start time of alarm instances a_1 and a_2, and the end time of instance a_1, respectively.

3. Calculate the probability that an instance of a_1 will trigger one or more a_2s, and vice-versa, where the probability of an instance of a_1 triggering one or more instances of a_2 is given as:

   $$\Pr(a_{1trig}a_2) = |a_{1trig}a_2|/|a_1| \tag{6.2}$$

4. From here, the relationship between the two alarms will be determined as follows:

   (a) If $\Pr(a_{1trig}a_2) >= 0.7$ and $\Pr(a_{2trig}a_1) >= 0.7$, then alarms a_1 & a_2 usually appear together

   (b) If $\Pr(a_{1trig}a_2) <= 0.2$ and $\Pr(a_{2trig}a_1) <= 0.2$, a_1 & a_2 never or rarely appear together

   (c) If $\Pr(a_{1trig}a_2) >= 0.7$ and $\Pr(a_{2trig}a_1) <= 0.2$, a_2 will usually be triggered whenever alarm a_1 appears; a_2 is a more general alarm

   (d) If $\Pr(a_{1trig}a_2) <= 0.2$ and $\Pr(a_{2trig}a_1) >= 0.7$, a_1 will usually be triggered whenever alarm a_2 appears; a_1 is a more general alarm

   (e) If none of the above, the two alarms are randomly or somewhat related

The results of this allow alarms related to different aspects of the chosen assembly to be identified, which will be analysed in the next step. Alarms related

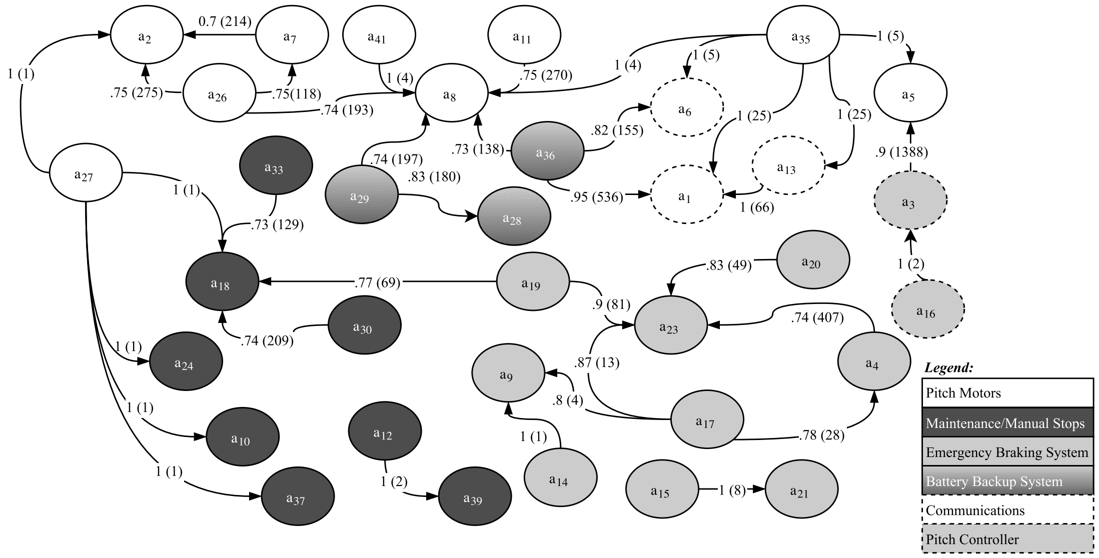


Figure 6.3: Network diagram of pitch system alarm triggers, grouped according to the sub-assembly which they belong to. The arrows are labelled with the probability one alarm will trigger another, and the absolute number of times this happened shown in brackets

to specific auxiliary systems or manual intervention which are usually *reactions* to faults which have occurred in the assembly being focused on, or are simply not relevant, can be excluded. This step can be made easier by graphically analysing the results of the probability analysis in a network diagram. The final set of k alarm codes obtained is referred to as A_r:

$$A_r = [a_1, \ldots, a_k] \tag{6.3}$$

In this case, the probability-based analysis was performed on the 58 alarms identified in the previous step. A network diagram showing the relationships between these alarms was then constructed, as seen in figure 6.3. An alarm with an arrow leading from it to another alarm indicates that it usually triggers the other alarm. The numbers labelled along the arrows show the probability of one alarm triggering another, with the absolute number of times the alarm was triggered shown in brackets. Alarms which were not shown to generally trigger other alarms were left out of this diagram. This allows alarms to be attributed to various sub-assemblies or functions within the relevant assembly, where it is not clear from the documentation. As can be seen, there are a number of different "groups" of alarms, related to different sub-assemblies within the pitch system.

The alarms to be analysed in the next step were selected according to the following criteria:

- The alarm causes the wind turbine to stop generating
- The alarm is not related to a reaction to another alarm, e.g. safety-chain or maintenance-related alarms
- There were enough instances (> 25) of the alarm for it to be analysed effectively

Alarms related to the battery backup and emergency braking systems, and alarms related to manual control or maintenance were excluded, as well as alarms which do not cause the turbine to stop generating (e.g. information messages related to system tests). Of the remaining alarms, only those with enough instances for useful analysis were included, i.e. >25 instances. This number was found heuristically by iterating through this methodology to find a good balance of including relevant alarms without introducing too much noise caused by very rarely occurring alarms. This resulted in a set of 30 alarms relevant to faults in the pitch system.

Alarms which represent the same fault, but, for example, on a separate turbine blade axis, were given the same shared alarm code. This was to ensure that alarm sequences along different axes would be grouped together as the same type of fault. For example, alarm codes a_{18}, a_{19} and a_{20} represent "Pitch Control Deviation on Axis x fault", where x is 1, 2 or 3, respectively. All these alarm codes have been renamed a_{18}, to group them together as one. There were 12 of these "duplicate" alarms, to give a final set of 18 relevant alarms for further analysis:

$$A_r = [a_1, a_2, \ldots, a_{18}] \tag{6.4}$$

6.3.2 Group similar sequences of alarms

6.3.2.1 Create "batches" of relevant alarm sequences

Before clustering, "batches" of fault alarm sequences which occur during stoppages must be identified. The first step is to identify the alarm code that signifies the turbine returning to normal operation. This is usually an information alarm to communicate that the turbine has been brought back on-line after a fault alarm-related stoppage. In this case this code is referred to as a_n. Its associated description was "returning to normal operation". Once this has been found, the next step is to create "batches" of alarm sequences associated with each turbine,

Table 6.1: Example of a batch of alarm sequences (all alarms belong to same turbine)

t_s	Code	Description
24/12/2015 09:05:40	a_1	Blade angle asymmetry
	a_2	Pitch thyristor fault
24/12/2015 09:05:52	a_4	Blade braking time too high
24/12/2015 09:06:22	a_1	Blade angle asymmetry
	a_3	Pitch control deviation
	a_5	Pitch malfunction 2 or 3 blade
24/12/2015 09:06:57	a_5	Pitch malfunction 2 or 3 blade

i.e. the alarm instances in each batch must all belong to the same turbine.

Each batch is created as in algorithm 3. This process is similar to the one outlined in section 4.3, with two key differences. First, the resulting batches here only contain alarms from A_r, as opposed to any alarms which occurred between the turbine stopping/being curtailed, and coming back on-line. This is because the sequences in question must be directly related to faults in the assembly under consideration, and not reactions to it. Secondly, previously, batches on the same turbine which occurred within one hour of each other were joined together as one continuous batch. That is not the case here, as the faults are being analysed with more granularity and individual sequences will hold more importance.

The results of this were a total of 456 batches of alarm sequences representing 456 individual stoppages across all 11 turbines in the 12 months of data. A typical example of a batch can be seen in table 6.1. It should be noted here that the final a_n alarm itself is not included in the batch at the analysis stage, but is provided when displaying batches so as to see how long the stoppage lasted. As can be seen, there are a mixture of alarms which occur individually and simultaneously (sharing common t_ss) to give a total number of 7 alarm instances with with four different t_ss.

6.3.2.2 Extract features from these batches

In order for the clustering to be effective, useful features from the alarm sequences must be extracted. Three separate ways of extracting feature vectors

Algorithm 3: Create batches of alarm sequences of relevant alarms

```
Data: Return to normal operation alarm, $a_n$; Relevant alarms $A_r$; All alarm
      instances $L$; Set of turbine numbers $T$
1 for $j$ in $T$ do
2     Find all instances of $L$ from turbine $j$. Store set of resultant instances as
        $L^j$
3     Find all alarm instances from $L^j$ which have a code in $A_r$. Store set of
        resultant instances as $L^j_r$
4     Find all instances from $L^j$ which have code $a_n$. Store set of resultant
        instances as $L^j_n$
5     Find earliest occurring instance in $L^j_r$. Store its $t_s$ as $t_{b\_start}$
6     Find earliest occurring instance in $L^j_n$ with $t_s > t_{b\_start}$. Store its $t_s$ as
        $t_{b\_end}$
7     Create a batch of alarms, $B_i$, from $L^j_r$ with $t_s$ that satisfy:

                    $t_{b\_start} \geq t_s < t_{b\_end}$                    (6.5)

8     Find earliest occurring instance in $L^j_r$ with $t_s > t_{b\_end}$. Store its $t_s$ as
        $t_{b\_start}$
9     Repeat steps 7 - 9 until no more instances of $L^j_r$
```

for each sample, F_1, F_2 and F_3, are explored.

F_1 - Base Case

The first feature extraction method was based solely on the order the alarms appeared in each batch. Batches can have a varying number of alarms, but in order to stop outlier batches with a disproportionately large number of alarms influencing the clustering algorithm, only batches with up to a certain maximum number of alarm instances, m_a, are included. This is because these outliers would likely not be included in any cluster, or placed in single clusters where each would be the sole member.

The feature vector for a batch, F_1, consists of a vector of 0s of length $k{*}m_a$, with a 1 being placed in the relevant location indicating the presence of an alarm:

$$F_1 = [f^1_1, f^1_2, \ldots, f^1_k, \ldots, f^{m_a}_1, f^{m_a}_2, \ldots, f^{m_a}_k]^T \tag{6.6}$$

A simplified example can be seen in figure 6.4, where there are four possible alarm codes, and a maximum of three alarm instances, i.e. $k = 4$, $m_a = 3$ and $A_r = [a_1, a_2, a_3, a_4]$. In batch 1, the first alarm is a_3, so a 1 is placed at f^1_3. The

$$\begin{array}{ll} \multicolumn{2}{c}{\textit{Batch 1}} \\ \hline t_s & \text{Code} \\ \hline \text{13:07:13} & a_3 \\ & a_1 \\ \text{13:10:48} & a_4 \end{array} \rightarrow F = \left[\begin{array}{c}0\\0\\1\\0\\\hline 1\\0\\0\\0\\\hline 0\\0\\0\\1\end{array}\right] ; \quad \begin{array}{ll} \multicolumn{2}{c}{\textit{Batch 2}} \\ \hline t_s & \text{Code} \\ \hline \text{15:02:00} & a_2 \end{array} \rightarrow F = \left[\begin{array}{c}0\\1\\0\\0\\\hline 0\\0\\0\\0\\\hline 0\\0\\0\\0\end{array}\right]$$

Figure 6.4: Simplified examples of F_1 construction

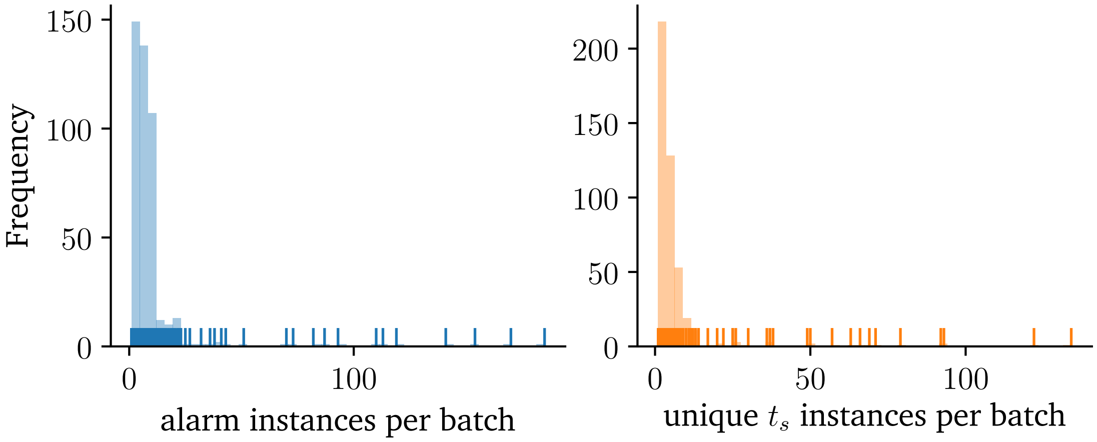


Figure 6.5: Histogram showing the distribution of number of alarm instances and unique t_ss in each batch. On the x-axis is a rug plot showing the location of the individual samples which make up each histogram

second alarm is a_1, so a 1 is placed at f_1^2. The third and final alarm is a_4, so a 1 is placed at f_4^3. In batch 2, there is only one alarm, a_2, so a 1 is placed at f_2^1. Note that the horizontal lines in the vector here are just to make the example easier to interpret.

Figure 6.5 shows the distribution of number of alarm instances and unique t_ss per batch in the dataset used in this study. As can be seen, over 90% of batches had between 1 and 20 alarm instances, so in this case $m_a = 20$ was selected as the maximum number of alarm instances. This led to 425 batches with an average of 6.75 alarms per batch. With $k = 18$ possible alarms, the length of

$$\begin{array}{ll} \multicolumn{2}{c}{\textit{Batch 1}} \\ \hline t_s & \text{Code} \\ \hline \text{13:08:02} & a_1 \\ & a_2 \\ & a_4 \\ \text{13:08:50} & a_3 \\ \text{13:12:10} & a_1 \\ & a_3 \end{array} \rightarrow F = \left[\begin{array}{c} 1\\1\\0\\1\\ \hline 0\\0\\1\\0\\ \hline 1\\0\\1\\0 \end{array}\right] ; \quad \begin{array}{ll} \multicolumn{2}{c}{\textit{Batch 2}} \\ \hline t_s & \text{Code} \\ \hline \text{15:02:30} & a_1 \\ & a_2 \end{array} \rightarrow F = \left[\begin{array}{c} 1\\1\\0\\0\\ \hline 0\\0\\0\\0\\ \hline 0\\0\\0\\0 \end{array}\right]$$

Figure 6.6: Simplified examples of F_2 construction

each feature vector F was $18 * 20 = 360$.

F_2 - Incorporating simultaneous start times

The batch in table 6.1 has a number of alarms occurring simultaneously. This is a similar case for many batches, so it was decided to extract a feature set that takes this into account by grouping alarms according to their t_s. Only batches with up to a certain maximum number of t_s, m_t, were included. Similarly to F_1, where only batches with up to a certain number of unique alarms were included, this was done to avoid issues with outliers.

The feature vector for a sample, F_2, once again consisted of a vector of 0s, this time of length $k * m_t$:

$$F_2 = [f_1^1, f_2^1, \ldots, f_k^1, \ldots, f_1^{m_t}, f_2^{m_t}, \ldots, f_k^{m_t}]^T \tag{6.7}$$

A simplified example is shown in figure 6.6, using $k = 4$, $m_t = 3$ and $A_r = [a_1, a_2, a_3, a_4]$. In the first batch, there are three alarms occurring at the first t_s (13:08:02), a_1, a_2 and a_4, so 1s are placed at f_1^1, f_2^1 and f_4^1. There is only one alarm, a_3 at the next t_s, so an alarm is placed at f_3^2. Two alarms, a_1 and a_3 occur at the final t_s, so 1s are placed at f_1^3 and f_3^3

As seen in figure 6.5, over 90% of batches have between 1 and 10 unique t_ss, so m_t was set to 10. This translated to 417 batches, with a mean of 6.57 alarms spread across 3.84 t_ss in each batch. Each feature vector was $18*10 = 180$ long.

F_3 - Incorporating the time between each t_s

The final feature extraction method expands on the previous method by incorporating the time between each t_s, representing how long the alarms at that t_s persisted before other alarms were triggered. It does this by making two slight changes to F_2. First, the time in seconds between each $t_s$is added as an extra feature at the end of each group of k alarms seen in F_2. In the case of the last alarm, the time between its t_s and the t_s of the "returning to normal operation" alarm for that batch, a_n, is used. This means the final length of the vector is $(k + 1) * m_t$.

Because the new features can be $\gg 1$, there is a chance the clusters could be heavily biased towards grouping batches with similar numbers of t_s, and the time between these t_ss, without taking into account the actual alarm codes themselves. Hence, the second change is that different values can be substituted for 1, such as 100, 1000, etc.

An example is provided in figure 6.7. This is identical to the example in the last section, but with the extra features added. Note that in this example $X = 100$, so 1s are replaced with 100s. In the first batch, the first "extra" feature is 48, signifying the time difference in seconds between the first t_s (13:08:02) and the second t_s (13:08:50), so this is placed at position f_5^1. The fourth and final t_s is 13:12:10, and the t_s of a_n is 13:15:10. This is 180s after the final t_s, so 180 is placed at position F_5^4. Note that the a_n alarms are included here only to show where the final "extra" feature comes from; they are not included in batches during cluster analysis.

As before, batches with between 1 and 10 unique t_ss are used, for a total of 417 batches. Each feature vector this time was $(18 + 1) * 10 = 190$ long.

6.3.2.3 Perform cluster analysis

Identifying patterns in high-dimensional data with no "ground truth" to learn from is an unsupervised learning problem (James et al. 2013). Cluster analysis is a powerful unsupervised learning technique that is used to identify patterns in samples of data and group samples with similar patterns together, so is ideally suited to this problem (Hastie et al. 2009). The main goal of clustering

$$\begin{array}{ll} \multicolumn{2}{c}{\textit{Batch 1}} \\ \hline t_s & \text{Code} \\ \hline 13{:}08{:}02 & a_1 \\ & a_2 \\ & a_4 \\ 13{:}08{:}50 & a_3 \\ 13{:}12{:}10 & a_1 \\ & a_3 \\ \textit{13:15:10} & a_n \end{array} \rightarrow F = \left[\begin{array}{c} 100\\ 100\\ 0\\ 100\\ 48\\ \hline 0\\ 0\\ 100\\ 0\\ 200\\ \hline 100\\ 0\\ 100\\ 0\\ 180 \end{array}\right]; \quad \begin{array}{ll} \multicolumn{2}{c}{\textit{Batch 2}} \\ \hline t_s & \text{Code} \\ \hline 15{:}02{:}30 & a_1 \\ & a_2 \\ \textit{15:04:35} & a_n \end{array} \rightarrow F = \left[\begin{array}{c} 100\\ 100\\ 0\\ 0\\ 125\\ \hline 0\\ 0\\ 0\\ 0\\ 0\\ \hline 0\\ 0\\ 0\\ 0\\ 0 \end{array}\right]$$

Figure 6.7: Simplified examples of F_3 construction

is to group a collection of objects into separate subsets or "clusters", so that the objects in each cluster are more similar to each other than those in other clusters.

Agglomerative clustering is a type of hierarchical, or tree-based, clustering which is well suited to data that has a large number of clusters. In this case, there may be many different alarm sequences so this is an appropriate technique to use. It starts by assigning every individual sample into its own unique cluster. It then looks at the pairwise similarity between all the clusters, and merges the two which are most similar to each other. It repeats this process until some specified number of clusters remain. The result of this is a binary tree linking each sample into one of a number of clusters (Hastie et al. 2009). In this work, the Euclidean distance between the centre of each cluster is used as a similarity metric.

Density-based spatial clustering of applications with noise (DBSCAN), is another powerful clustering technique that does not need many parameters and automatically decides on an optimal number of clusters. DBSCAN views clusters as areas of high density (i.e. many samples in close proximity to each other) separated by areas of low density. It does this by first assigning some samples as "core" samples. These are defined as samples which have a certain minimum number of "neighbours", with neighbours being defined as samples within some minimum amount of distance to them. Clusters are built by recursively selecting a core sample, finding all of its neighbours which are core samples, finding all of their neighbour core samples, and repeating until there are no further neighbour core samples within that cluster. All other samples which are not core samples (i.e. do not meet the minimum number of neighbours), but which are themselves neighbours of a core sample, are assigned to that core sample's cluster. Any samples which are not in the neighbourhood of any core sample are classed as outliers, and not assigned a cluster. In this way, DBSCAN decides on its own optimal number of clusters (Ester et al. 1996). Because DBSCAN automatically decides on an optimal number of clusters, it was decided to compare this with agglomerative clustering in this work.

Clustering performance was evaluated in two ways. First, the silhouette coefficient was used as a measure of how well defined the clusters are. The silhouette coefficient is defined as follows:

$$s = \frac{b - a}{max(a, b)} \tag{6.8}$$

where a is the mean distance between a sample and all other samples in the same cluster, and b is the mean distance between a sample and all other points in the next nearest cluster. The silhouette coefficient takes a value between -1 and 1, with 1 meaning the point is far away from its neighbouring cluster, 0 meaning it's on the boundary, and -1 meaning the point has possibly been misclassified. The silhouette coefficient is hence a score given to every sample in a cluster and is evaluated graphically, as will be seen in section 6.4.

The silhouette coefficient is a good indication of how well the clustering is performing, but only if the features that have been extracted accurately represent the underlying data. For this reason, in some cases, a manual inspection of the clusters was performed to ensure that good/bad silhouette scores translated to effective clustering for this specific use case. The manual inspection involved selecting 2-3 samples from each cluster and checking if the alarm sequences in each sample were similar to each other if there was a high silhouette score, or dissimilar for a low silhouette score.

The agglomerative clustering and DBSCAN algorithms were applied to the three different feature sets extracted in the previous step, F_1, F_2 and F_3, with F_3 being trained with various different values of $X; X \in \{1, 10, 100, 1000\}$. Since agglomerative clustering takes a specific number of clusters as an input, it is normally trained several times with a number of different clusters to find the optimal number. Here, the analysis was carried out for between 2 and 20 clusters, with the optimal number of clusters being selected according to the one with the highest silhouette coefficient.

Once the optimum number of clusters has been found for agglomerative clustering, it is evaluated against DBSCAN. The final evaluation is performed again using the silhouette score, with the effectiveness of the score being checked via manual inspection. The most effective method is decided heuristically, e.g. if agglomerative clustering manages to group 40% of clusters with reasonable accuracy, but DBSCAN classifies 30% of clusters with perfect accuracy, then DBSCAN in that case would be a better choice. The results of this can be found in the following section.

6.4 Clustering Results

As described in section 6.3.2.1 (step 2.1 from figure 6.1), 456 batches were created from the full set of data. Three sets of features, F_1, F_2 and F_3 were extracted from these batches, as described in section 6.3.2.2 (step 2.2 of figure 6.1). With m_a set to 20 and m_t set to 10, this meant there were 425 samples of F_1 and 417 samples of F_2 and F_3. The results of applying the clustering method described in section 6.3.2.3 (step 2.3 of figure 6.1) are discussed in this section.

In all cases, both silhouette and manual analyses were performed. A summary of the results can be seen in table 6.2. This table shows the name, no. of clusters, silhouette score, and % of samples which achieved a silhouette score of >0.9. The table also shows whether or not the silhouette score gave a good indication of accurate clustering, as determined from manual inspection. Note $X = 1$ is the only one included for feature set 3 as this was the best scoring value of X.

Table 6.2: Results Summary

Feature Set - Algo.	No. Clusters	Avg. Sil.	$\% > .9$	Accurate Sil.?
1 - Agg	20	.2	16	Y
1 - DBSCAN	13	1	27.3	Y
2 - Agg	20	.39	3.8	Y
2 - DBSCAN	15	1	45.1	Y
3 - Agg ($X = 1$)	8	.82	91.4	N
3 - DBSCAN ($X = 1$)	7	.93	41.2	N

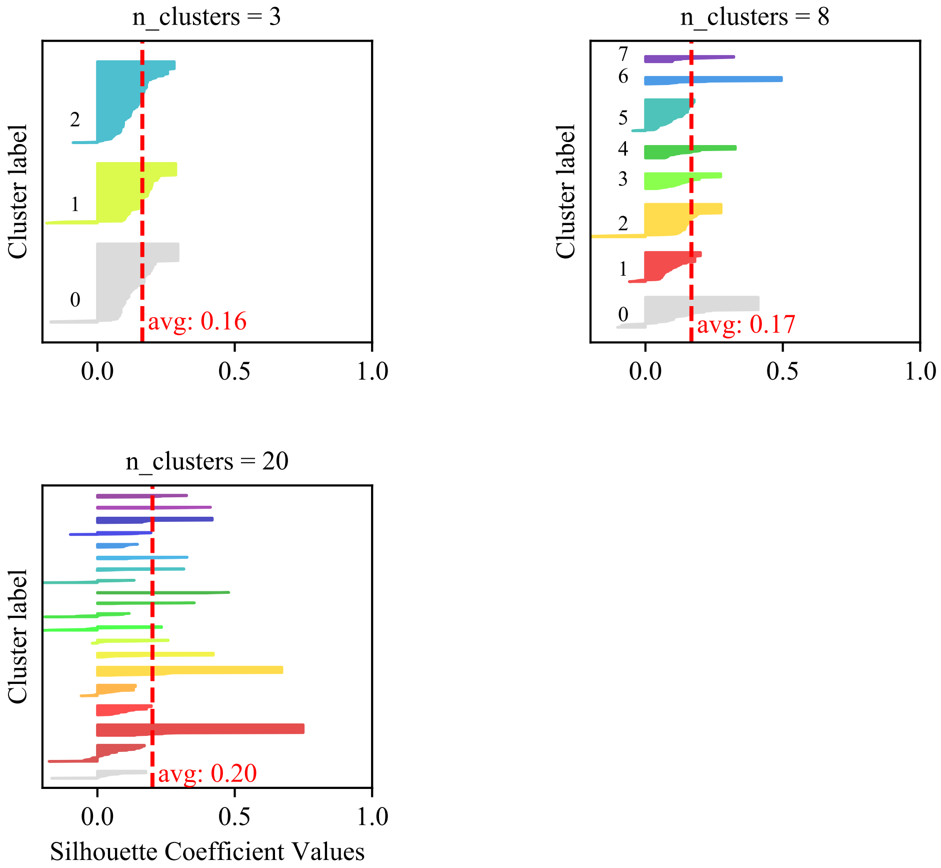


Figure 6.8: Silhouette analysis for agglomerative clustering using feature set 1. Thickness of the bars indicates number of samples in that cluster

6.4.1 F_1 – Base Case

6.4.1.1 Agglomerative Clustering

Silhouette analysis was carried out for the agglomerative clustering with between 2 and 20 clusters. The silhouette analysis for the case of 3, 8 and 20 clusters can be seen in figure 6.8.

In this figure, each cluster label represents the silhouette scores of every batch sample in that cluster, sorted in increasing order. This means that the thicker the silhouette plot for each cluster, the more samples there are in that cluster.

As can be seen, a higher number of clusters in agglomerative clustering performed better. The best average silhouette score was found on the maximum 20 clusters, with an average silhouette score of .2. However, there were big fluctuations in the silhouette scores of members within each cluster. Manual

Table 6.3: Samples within a high scoring cluster for DBSCAN using feature set 1

t_s	Code	Description
	Sample 1	
2016-05-05 13:08:42	a_6	Pitch controller comms fault
2016-05-05 13:08:55	a_5	Pitch malfunction 2 or 3 blade
	Sample 2	
2015-06-26 12:25:11	a_6	Pitch controller comms fault
	a_5	Pitch malfunction 2 or 3 blade

inspection confirmed that the clusters that scored well contained batches that had very similar alarm sequences, however the clusters that scored above 0.9 represented only 16% of samples fed into the algorithm.

6.4.1.2 DBSCAN

DBSCAN in this case classified 27.3% of samples into 13 different clusters. The average silhouette score across all clusters was 1. A manual inspection confirmed that the sequences of alarms in samples within each cluster were identical in nearly all cases.

An important point to note is that because the F_1 features did not take into account whether some alarm instances appeared simultaneously, in a small number of cases there were different numbers of t_ss in each sample within a cluster. An example of this can be seen in table 6.3, showing two samples from the same cluster. In *Sample 1* the 2 alarm instances happen in sequence, whereas in *Sample 2*, they happen simultaneously. This can be relevant as the root cause related to the alarm sequence in both cases could possibly be different; in *Sample 1* there was a pitch malfunction in the blades (i.e. the pitch angles in all three blades were not equal), which was caused by a communication fault in the pitch controller. In *Sample 2* the two occurred simultaneously, which in cases with more complex alarm sequences could point to different root causes.

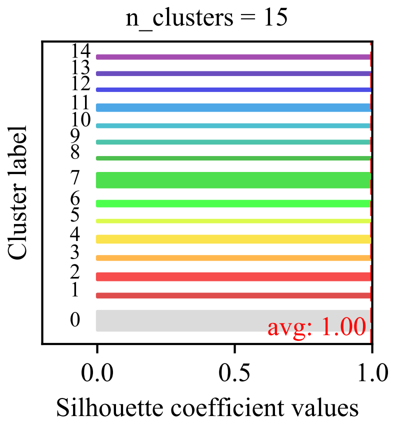


Figure 6.9: Silhouette scores for DBSCAN using F_2 features. Thickness of the bars indicates number of samples in that cluster

6.4.2 F_2 – Incorporating Simultaneous Start Times

6.4.2.1 Agglomerative Clustering

Here, the optimum number of clusters for agglomerative clustering was again found to be 20, with a silhouette score of 0.39. Only 3.8% of samples were clustered with a score above 0.9.

6.4.2.2 DBSCAN

DBSCAN once again performed much better than agglomerative clustering, with an average silhouette score of 1 across 15 clusters, as seen in figure 6.9. This represented 45.1% of samples fed into the algorithm. A manual investigation of the clusters revealed that not only were the alarm sequences in each sample within clusters identical, but each sample also had the same number of t_ss, i.e. the information about alarm instances that occurred simultaneously was preserved.

6.4.3 F_3 – Incorporating the Time Between Each t_s

The above analysis was repeated using the new time-based features for various values of X.

For this analysis, the feature array was normalised before clustering, to avoid large times between t_ss having a disproportionate impact.

6.4.3.1 Agglomerative Clustering

The optimal number of clusters across all values of X was found to be 8. With 8 clusters, the highest average silhouette score was 0.85, for $X = 1$. However, a manual inspection of the clusters showed that the samples in each varied wildly. This was probably down to the fact that setting $X = 1$ means the clustering barely takes into account the actual alarms that were generated, and focuses almost solely on the times between each t_s in a batch, which could be much greater than 1. Even after normalisation, the average value of features representing these times between each t_s was 0.331, whereas the value of the features representing the presence of a particular alarm code (i.e. the features which are marked as "1" for $X = 1$) was 0.001.

For $X = 10$, $X = 100$ and $X = 1000$, silhouette scores were 0.82, 0.52, and 0.26, respectively. However, once again the batches in each cluster were quite different. With manual inspection, it was found that in batches with identical alarm sequences, there was a wide range of possible values for the time between each t_s, i.e. even though the sequences of alarm instances in two different batches could be identical, the time between these alarms could considerably vary. This could mean that effective clustering using these extra features was not possible.

6.4.3.2 DBSCAN

DBSCAN produced 7 clusters for all values of X, with silhouette scores of 0.93, 0.91, 0.8 and 0.43 for $X = 1$, $X = 10$, $X = 100$ and $X = 1000$, respectively. Once again manual inspection showed that there was wide variation in the samples within each cluster. This added further evidence to the fact that the extra features created for feature set 3 were not suitable for effective clustering.

6.4.4 Analysis of Results

Based on the above results, DBSCAN performed on the F_2 features yielded the best results. This resulted in 15 clusters of batches, with each batch containing

Table 6.4: Example of alarm sequences found in cluster 2

t_s	Code	Description
2015/06/29 14:44:29	a_1	Blade angle asymmetry
	a_2	Pitch thyristor fault
2015/06/29 14:44:34	a_4	Blade braking time too high
2015/06/29 14:44:37	a_1	Blade angle asymmetry
	a_3	Pitch control deviation
	a_5	Pitch malfunction 2 or 3 blades
2015/06/29 14:44:38	a_3	Pitch control Deviation
	a_5	Pitch malfunction 2 or 3 blades
2015/06/29 14:52:10	a_n	*Returning to normal operation*

an identical sequence of alarms. 45.1% of batches fed into the clustering algorithm were successfully assigned a group, which represented 41% of the 456 total batches analysed in the study. These correctly clustered batches together represented over 134 hours of downtime on the turbine, with each stoppage lasting an average of just over 43 minutes.

A sample of a batch from cluster 2 can be seen in table 6.4, showing the progression of a fault in the pitch system. Once again, the alarm a_n here is just provided to show how long the stoppage lasted in total. First, a fault in the thyristor of one of the pitch motor circuits is detected, which simultaneously causes asymmetry in the pitch angles across the three blades. Because of this, the turbine is not braking quickly enough, which sets off the a_4 alarm, as well as blade angle asymmetry alarms for the other blades, a pitch control deviation alarm and a more "general" alarm showing a pitch malfunction across more than one blade. The other batches in this cluster showed the exact same alarm sequence, including which alarms occurred simultaneously.

Overall, these results show that a large proportion of the alarm sequences which occur during individual stoppages associated with the pitch system can be accurately sorted into a number of distinct groups. The implications of this will be discussed in the following section.

6.5 Conclusions

This work focused on attempting to sort similar sequences of alarms as they occurred during wind turbine stoppages into several distinct groups, with the aim of reducing the burden of analysis on turbine operators when high volume alarm showers are generated. The alarms generated during 456 different stoppages were analysed. Sequences of alarms as they occurred during each stoppage were identified, with each "batch" of alarm sequences being associated with a particular stoppage. Three different sets of features representing the alarms in each batch were extracted, and clustering techniques applied with the aim of grouping similar batches together. The first feature set looked solely at the order the alarms appeared in. The next set took into account whether or not some alarms occurred simultaneously. The third feature set took into account the time between the alarms in each batch. Two different clustering techniques, agglomerative clustering and DBSCAN, were applied to these three feature sets, and the results of each compared.

The results for the first feature set showed promise, with DBSCAN managing to accurately cluster 27.4% of samples, representing over 110 hours of downtime. A drawback was that the samples within each cluster did not take into account whether some alarm instances occurred simultaneously or not. Agglomerative clustering in this case showed poor results. The best results occurred on the second feature set using DBSCAN; 45.1% of batches were accurately sorted into fifteen distinct clusters, which together represented over 134 hours of downtime on the turbine. In this case, whether or not some alarms occurred simultaneously was consistent within batches. Agglomerative clustering once again did not perform as well as hoped. The third feature set showed poor results all round, possibly due to there being too much variance of possible values for the time between alarm instances.

Based on these results, it is indeed possible to usefully group together similar sequences of alarm instances into distinct clusters. This means that the burden of analysis for turbine operators during stoppages can be reduced. If a stoppage occurs during live operation, and the resulting sequence of alarms can be attributed to a previously identified group of similar alarm sequences which occurred during past stoppages, the operator can be given information about the shared characteristics of these stoppages rather than seeing a cascade of individual alarms which need to be analysed. This information can be related to what corrective action, if any, was generally taken in the past, the severity of the fault and duration of associated down time, the root cause or other information to help diagnose the fault, whether the stoppage was controller-related, or others. As well as this, the frequency of particular alarm sequences can be tracked, which can give more information and context than simply tracking the frequency of individual alarms.

Chapter 7

Conclusions and Future Research

7.1 Summary of Research

This thesis has investigated various aspects of CM on wind turbines using SCADA data, i.e. alarm system and 10-minute operational and availability data. In chapter 2, the motivations for CM were given, highlighting in particular its usefulness in the offshore context. Also described were the advantages of leveraging already existing SCADA data on the turbine over retrofitting dedicated CM sensors, and the added functionality such a strategy could entail over the prevailing strategies used in the wind industry today. A review of CM methods found in the literature was presented, and classification was highlighted as an area with potential for prognostics that has not seen much focus, despite having a number of advantages over methods such as NBM and trending. Also reviewed were methods which attempt to curtail the information overload to operators associated with the high volume of SCADA alarms generated during fault events. This highlighted that additional fault diagnostic functionality can be gleaned through the analysis of turbine alarm sequences.

To this end, the research in this thesis can be split into two broad areas: that which focused on fault prognostics through classification techniques based on SCADA data, and research which focused on fault diagnostics through analysis of the turbine alarm system.

The first part, fault prognosis, was covered by chapters 3, 4 and 5. In chapter 3, a case study was performed where various levels of CM were attempted, leading up to limited fault prognosis. This chapter addressed some of the common pitfalls which have been seen in similar works in literature, and showed that fault prognosis is indeed possible using classification techniques, but that further investigation was warranted to improve performance. In chapter 4, a more granular method of generating a historical database of fault data from alarm system data was presented, with the aim of using this data for future use by classification-based CM techniques. A secondary benefit of this method was that the resulting database could be used for reliability analysis of various turbine assemblies. The work from these chapters was used to build up to a comprehensive prescriptive framework which was developed in 6. This robust methodology for classification-based fault prognostics additionally included a way to evaluate such a system as it would perform deployed in the field.

The second broad area of research, fault diagnostics based on the alarm system, was presented in chapter 6. Here, a methodology was developed for reducing the amount of raw data presented to operators during fault events. This provided a framework for reducing the burden of analysis on technicians when dealing with turbine alarms, which, as mentioned at the start of the chapter, can often be ineffective in their current form.

7.2 Research Objectives

This thesis outlined a number of objectives in chapter 1. They are presented here again along with the associated results which showed how these objectives were met.

RO1. Determine what level of condition monitoring can be performed using classification techniques

In chapter 3, the case study showed that fault detection was possible with high recall, but low precision using classification techniques. For fault diagnosis, the score was increased for some types of fault. In particular, generator heating faults could be diagnosed with an overall F1 score of .8. For fault prognosis, those same generator heating faults could be predicted with a maximum PH of 0-2 hours with an F1 score of .6. However, when this was extended out to a maximum PH of 24-48 hours, F1 score dropped below .15. These results showed that CM up to the level of fault diagnosis could be effectively performed, but more work needed to be done for effective fault prognosis.

A more robust methodology was developed in chapter 5 with some notable improvements to the labelling process, and which took into account findings from chapter 3 with regards to the sampling methods used for the training data. Here, a precision score of .155 and recall of .49 was achieved (equating to an F1 score of .24), at a PH of 6-48 hours. Although these represented quite low classification scores, a sliding window metric was developed which took into account the number of fault-classified samples in a certain period. This metric showed that effective prognosis was possible, with the ratio of caught faults to false alarms being configurable through tweaking of certain parameters.

The findings from these chapters showed that CM at the level of fault prognosis is possible using classification techniques, though more work is needed to be done to further improve scores.

RO2. Investigate different techniques for dealing with classification based on imbalanced datasets and evaluate their suitability for fault detection, diagnosis and prediction

Chapter 3 investigated a number of techniques for dealing with the inherently massive imbalance seen in training data for CM. A variety of techniques were investigated at all three levels of CM, i.e. fault detection, diagnosis and prognosis. These were: undersampling, adding a class weight parameter to the minority class(es), easy ensemble, cluster centroids, edited nearest neighbours, SMOTE and Tomek Links. It was found that randomly undersampling produced the most consistent results across all levels of CM.

RO3. Determine whether information on historical failures can be accurately gleaned through analysis of the turbine alarms system, and whether this information can be used to create a complete and accurate training set for fault prediction

Chapter 4 showed a methodology for building a database of historical stoppages solely from turbine alarm and availability data. The stoppages were classified as being from one of a number of different categories, including grid or weather related faults, planned or unplanned maintenance and faults. For fault-related stoppages, the assembly the fault occurred in and whether the fault resulted in a site visit (either repair action or manual on-site reset) were also given.

The duration of all stoppages were also recorded. By cross-referencing with maintenance logs and availability data, it was shown that the resulting database was accurate in all respects. This data was overlaid on an associated set of 10-minute SCADA data in chapter 5 to build an accurate set of training data for classification.

RO4. Design a comprehensive framework which incorporates all previous findings as well as best practices from literature and apply this methodology to evaluate its performance as a field-deployed system

Chapter 5 incorporated findings from the case study in chapter 3 and the data labelling process in chapter 4 to build a formalised methodology for fault prediction. It included machine-learning best practices, and made recommendations to avoid some of the pitfalls seen in previous works which used a classification-based approach for CM on wind turbines. Part of this methodology included a sliding window metric for generating alerts when deployed in the field, and this was evaluated on a held-out test set. Samples were fed to the resulting system one-by-one, simulating field deployment. As mentioned in RO1, the results showed that such a system could be successfully used for CM, with parameters for configuring the ratio of missed faults to false alarms.

RO5. Investigate whether the burden of analysis on maintenance technicians during fault events can be effectively reduced by gleaning information from the high volume of generated alarms

In chapter 6, a system for identifying relevant sequences of alarms as they appear during fault events was developed. Similar sequences were then grouped together using clustering techniques, so that any shared characteristics between these clusters could be evaluated by maintenance technicians. Applying this to 12 months' of data across 11 turbines showed there were 456 different stoppages related to the pitch assembly. The clustering techniques managed to sort 45% of these stoppages into just 15 different clusters. This massively reduces the burden of analysis on maintenance technicians; if a sequence of alarms matching one of these clusters appears during a stoppage in live operation, the operator can be given information about the shared characteristics of these stoppages rather than seeing a cascade of individual alarms which need to be analysed.

7.3 Critical Analysis of Work and Future Recommendations

Here, a critical analysis of some of the work undertaken in this thesis is presented.

7.3.1 Chapters 3 - 5

At the outset of the PhD work which this thesis represents, a review of the existing literature was performed to identify gaps and opportunities for impactful research. Classification was highlighted as a technique for CM which has not seen much coverage in this domain in the literature, but which has high potential for very good results, given its success in applications in other domains in recent years (particularly through the use of deep learning). However, it quickly became clear that the limiting factor in this domain is availability and quality of data. These issues with data availability in the wind industry are discussed at various stages in the thesis, including sections 2.6.

This difficulty was first directly encountered in the work represented in chapter 3, where no granular maintenance logs were available, so faults were labelled with single instances of individual alarms. Furthermore, the data was related only to a single turbine over an 11-month period. Chapter 4 remedied the first of these issues by associating "batches" of alarms with particular types of stoppages, and this methodology was incorporated into the prescriptive framework developed in chapter 5. This led to increased classification scores, but these scores were still below the researcher's initial targets of $> .8$ for precision and recall. Incorporating a sliding window metric significantly remedied this, but catching a majority of faults still raised a number of false alarms (64% of all faults could be caught, but with > 100 hours of active false alarms per year).

A reason for these relatively low classification scores is posited to be because the faults being predicted were, although frequent, relatively minor (most resulting in stoppages which resulted in a remote reset after less than 30 minutes of down time). More severe faults, which lasted longer or required some kind of manual intervention, would be expected to be easier to detect due to leading fault signatures being more obvious for faults related to physical degradation of components. Although performance on these faults was slightly better, it was still not close to the goal of > .8 precision/recall scores. The reason for this was, once again, a lack of available data; 6 months of data across 11 turbines meant that there were simply not enough examples of faults for the classifiers to effectively learn what they looked like.

In domains such as image recognition, databases of millions of samples are used to train deep neural networks (He et al. 2016). In the classification performed in chapters 3 and 5, the maximum number of instances numbered in the hundreds. Hence, deep learning, which is computationally expensive and needs a large amount of data in order to realise its benefits, was not used. It is envisaged that by gaining access to a large amount of SCADA data, such as that owned by a utility-scale operator or OEM, the added benefits of classification-based CM over other SCADA-based methods such as NBM can be realised.

The framework developed in chapter 5 was designed to be as robust and comprehensive as possible, so that future researchers may apply it should they have access to such a large dataset. As mentioned elsewhere in this thesis, maintenance logs are not always stored in a structured, digital format, so any faults would have to be manually labelled in many cases. However, manually labelling a dataset comprising of hundreds of thousands of fault instances could be prohibitively tedious. Hence, the part of the framework developed in chapter 4, which deals with automatically building a historical failure database, was designed to specifically address this. Furthermore, as mentioned in section 2.8.4, many of the classification-based approaches seen in the literature saw some shortcomings which skewed the final results. Hence, this research should act as a guide to best practices to avoid some of these pitfalls going forward.

7.3.2 Chapter 6

As part of the work for dealing with alarm system data, maintenance technicians from two different operators were contacted. From liaising with these people, it was found that their experience with turbine alarm systems was consistent with past research discussed in other parts of this thesis; namely, while alarms can give a clue as to the reason a turbine has stopped generating, a lot of expert knowledge of the particular system is needed, and many alarms are simply ignored due to the volume generated. Hence, it was decided to try and reduce the burden of analysis for these technicians, and increase the utility of turbine alarm systems. The resulting methodology managed to successfully cluster 45% of all sequences of alarms into one of fifteen distinct clusters, where the sequences of alarms in each cluster were broadly similar.

The natural extension to this work involves investigating the different types of stoppages to identify their shared characteristics. Once these characteristics have been identified, not only can they be used for diagnosing future faults and deciding on the appropriate course of action post-occurrence, but can also be used for predictive purposes. If certain clusters are associated with specific faults in particular sub-assemblies or components, these can be used to train classifiers for an even more granular level of CM than that seen in chapter 5. With advance warning of theses specific types of faults, an appropriate course of action can be taken, with added knowledge of the shared set of characteristics that each cluster is likely to have.

7.4 Final Conclusions

This thesis presents a number of works which are intended to further advance the use of SCADA data for CM in wind turbines; through the use of classification for fault prognostics, and through cluster analysis of alarms for diagnostics. In this respect, the thesis has achieved these broad goals. However, it represents a first step towards implementing any of the solutions discussed, and not a final, field-ready solution. In this respect, it is hoped that future researchers in the space can use the work in this thesis, and indeed the publications which form part of it, as a platform to build upon with the ultimate goal of expanding the scope of CM possibilities using existing SCADA data.

Building a comprehensive and field-deployable fault prognostic or diagnostic system will, in the opinion of the author of this thesis, require a coordinated effort from stakeholders including operators and OEMs. As discussed by Kusiak in a recent *Nature* article in (Kusiak 2016), OEMs will need to be more open about sharing data and coordinating with researchers to further advance these goals. In the author's experience, signs of this are beginning to appear as the benefits are being communicated to OEMs, but the potential has yet to be fully realised.

References

Abbasion, S., Rafsanjani, A., Farshidianfar, A. & Irani, N. (2007), 'Rolling element bearings multi-fault classification based on the wavelet denoising and support vector machine', *Mechanical Systems and Signal Processing* **21**(7), 2933–2945.

Andrawus, J. A., Watson, J., Kishk, M. & Adam, A. (2006), 'The Selection of a Suitable Maintenance Strategy for Wind Turbines', *Wind Engineering* **30**(6), 471–486.

Arlot, S. & Celisse, A. (2010), 'A survey of cross-validation procedures for model selection', *Statistics Surveys* **4**, 40–79.

Bangalore, P., Letzgus, S., Karlsson, D. & Patriksson, M. (2017), 'An artificial neural network-based condition monitoring method for wind turbines, with application to the monitoring of the gearbox', *Wind Energy* **20**(8), 1421–1438.

Bangalore, P. & Tjernberg, L. B. (2013), An approach for self evolving neural network based algorithm for fault prognosis in wind turbine, *in* '2013 IEEE Grenoble Conference PowerTech, POWERTECH 2013'.

Bangalore, P. & Tjernberg, L. B. (2015), 'An artificial neural network approach for early fault detection of gearbox bearings', *IEEE Transactions on Smart Grid* **6**(2), 980–987.

Borchersen, A. B. & Kinnaert, M. (2016), 'Model-based fault detection for generator cooling system in wind turbines using SCADA data', *Wind Energy* **19**(4), 593–606.

Boser, B. E., Guyon, I. M. & Vapnik, V. N. (1992), A Training Algorithm for Optimal Margin Classifiers, *in* 'Proceedings of the Fifth Annual ACM Workshop on Computational Learning Theory', pp. 144–152.

Breiman, L. (1996), 'Bagging predictors', *Machine Learning* **24**(2), 123–140.

Breiman, L. (2001), 'Random Forests', *Machine Learning* **45**(1), 5–32.

British Petroleum (2017), 'BP Statistical Review of World Energy 2017', *British Petroleum* (66), 1–52.

Butler, S. (2012), Prognostic Algorithms for Condition Monitoring and Remaining Useful Life Estimation, PhD thesis, NUI Maynooth.

Butler, S., O'Connor, F., Farren, D. & Ringwood, J. V. (2012), A feasibility study into prognostics for the main bearing of a wind turbine, *in* 'Proceedings of the IEEE International Conference on Control Applications', Ieee, pp. 1092–1097.

Butler, S., Ringwood, J. & O'Connor, F. (2013), Exploiting SCADA system data for wind turbine performance monitoring, *in* 'Conference on Control and Fault-Tolerant Systems, SysTol', IEEE, pp. 389–394.

BVG Associates & WindEurope (2017), 'Unleashing Europe's offshore wind potential', (June), 64.

Cambron, P., Lepvrier, R., Masson, C., Tahan, A. & Pelletier, F. (2016), 'Power curve monitoring using weighted moving average control charts', *Renewable Energy* **94**, 126–135.

Carroll, J., McDonald, A. & McMillan, D. (2015), 'Reliability Comparison of Wind Turbines With DFIG and PMG Drive Trains', *IEEE Transactions on Energy Conversion* **30**(2), 663–670.

Carroll, J., McDonald, A. & McMillan, D. (2016), 'Failure rate, repair time and unscheduled O&M cost analysis of offshore wind turbines', *Wind Energy* **19**(6), 1107–1119.

Cawley, G. C. & Talbot, N. L. C. (2010), 'On Over-fitting in Model Selection and Subsequent Selection Bias in Performance Evaluation', *Journal of Machine Learning Research* **11**, 2079−2107.

Chang, C.-c. & Lin, C.-j. (2011), 'LIBSVM : A Library for Support Vector Machines', *ACM Transactions on Intelligent Systems and Technology (TIST)* **2**, 1–39.

Chawla, N. V., Bowyer, K. W., Hall, L. O. & Kegelmeyer, W. P. (2002), 'SMOTE: Synthetic minority over-sampling technique', *Journal of Artificial Intelligence Research* **16**, 321–357.

Chen, B., Matthews, P. C. & Tavner, P. J. (2013), 'Wind turbine pitch faults prognosis using a-priori knowledge-based ANFIS', *Expert Systems with Applications* **40**(17), 6863–6876.

Chen, B., Qiu, Y., Feng, Y., Tavner, P. & Song, W. (2011), Wind turbine SCADA alarm pattern recognition, *in* 'IET Conference on Renewable Power Generation (RPG 2011)', pp. 163–163.

Cohen, W. W. (1995), Fast Effective Rule Induction, *in* 'Machine Learning Proceedings 1995', Elsevier, pp. 115–123.

Colone, L., Reder, M., Tautz-Weinert, J., Melero, J., Natarajan, A. & Watson, S. (2017), Optimisation of Data Acquisition in Wind Turbines with Data-Driven Conversion Functions for Sensor Measurements, *in* 'Energy Procedia', Vol. 137, Elsevier B.V., pp. 571–578.

Conroy, N., Deane, J. & Ó Gallachóir, B. P. (2011), 'Wind turbine availability: Should it be time or energy based? – A case study in Ireland', *Renewable Energy* **36**(11), 2967–2971.

Cortes, C. & Vapnik, V. (1995), 'Support-vector networks', *Machine Learning* **20**(3), 273–297.

Cross, P. & Ma, X. (2014), 'Nonlinear system identification for model-based condition monitoring of wind turbines', *Renewable Energy* **71**, 166–175.

de Zeeuw, J. (1978), 'Peat and the Dutch Golden Age', *The historical meaning of energy attainability. AAG Bijdragen* **21**(1978), 3–31.

Department of Communications Climate Change and Environment (2010), National Renewable Energy Action Plan: Ireland, Technical report.

DNV GL (2017), Definitions of Availability Terms for the Wind Industry, Technical Report August.

Donovan, P. O., Leahy, K., Cusack, D. Ó., Bruton, K. & O'Sullivan, D. T. J. (2015), A data pipeline for PHM data-driven analytics in large-scale smart manufacturing facilities, *in* 'Annual Conference of the Prognostics and Health Management Society 2015', Vol. 6, pp. 1–10.

Du, M., Tjernberg, L. B., Ma, S., He, Q., Cheng, L. & Guo, J. (2016), 'A SOM based Anomaly Detection Method for Wind Turbines Health Management through SCADA Data', *International Journal of Prognostics and Health Management* **7**, 1–13.

Duffuaa, S. O. & Ben-Daya, M. (2009), *Handbook of Maintenance Management and Engineering*.

Eirgrid (2017), Industry Guide to the I-SEM, Technical report, EirGrid Group, Dublin.

Ester, M., Kriegel, H.-P., Sander, J. & Xu, X. (1996), A density-based algorithm for discovering clusters in large spatial databases with noise., *in* 'Knowledge Discovery and Data Mining', pp. 226—-231.

Esteva, A., Kuprel, B., Novoa, R. A., Ko, J., Swetter, S. M., Blau, H. M. & Thrun, S. (2017), 'Dermatologist-level classification of skin cancer with deep neural networks', *Nature* **542**(7639), 115–118.

European Parliament (2009), Directive 2009/28/EC of the European Parliament and of the Council of 23 April 2009, Technical Report 16.

Faulstich, S., Durstewitz, M., Hahn, B., Knorr, K. & Rohrig, K. (2008), Windenergy Report Germany 2008, Technical report.

Faulstich, S., Hahn, B. & Tavner, P. J. (2011), 'Wind turbine downtime and its importance for offshore deployment', *Wind Energy* **14**(3), 327–337.

Feng, Y., Qiu, Y., Crabtree, C. J., Long, H. & Tavner, P. J. (2011), Use of SCADA and CMS signals for failure detection and diagnosis of a wind turbine gearbox, *in* 'European Wind Energy Conference and Exhibition 2011, EWEC 2011', pp. 17–19.

Feng, Y., Tavner, P. J. & Long, H. (2010), 'Early experiences with UK round 1 offshore wind farms', *Proceedings of the Institution of Civil Engineers - Energy* **163**(4), 167–181.

FISHER, R. A. (1936), 'THE USE OF MULTIPLE MEASUREMENTS IN TAXONOMIC PROBLEMS', *Annals of Eugenics* **7**(2), 179–188.

Foley, A. M., Ó Gallachóir, B. P., McKeogh, E. J., Milborrow, D. & Leahy, P. G. (2013), 'Addressing the technical and market challenges to high wind power integration in Ireland', *Renewable and Sustainable Energy Reviews* **19**, 692–703.

Freund, Y. & Schapire, R. (1995), 'A desicion-theoretic generalization of online learning and an application to boosting', *Computational Learning Theory* **55**(1), 119–139.

Gallagher, C., Bruton, K., Leahy, K. & O'Sullivan, D. (2018), 'The suitability of machine learning to minimise uncertainty in the measurement and verification of energy savings', *Energy and Buildings* **158**.

Gallagher, C. V., Leahy, K., O'Donovan, P., Bruton, K. & O'Sullivan, D. T. (2018), 'Development and application of a machine learning supported methodology for measurement and verification (M&V) 2.0', *Energy and Buildings* **167**, 8–22.

García Márquez, F. P., Tobias, A. M., Pinar Pérez, J. M. & Papaelias, M. (2012), 'Condition monitoring of wind turbines: Techniques and methods', *Renewable Energy* **46**, 169–178.

Garlick, W. G., Dixon, R. & Watson, S. J. (2009), A Model-based Approach to Wind Turbine Condition Monitoring using SCADA Data, *in* '20th International Conference on Systems Engineering', Vol. 44, p. 7.

Godwin, J. L. & Matthews, P. (2013), 'Classification and Detection of Wind Turbine Pitch Faults Through SCADA Data Analysis', *International Journal of Prognostics and Health Management* **4**, 11.

Godwin, J. L. & Matthews, P. (2014), Rapid labelling of SCADA data to extract transparent rules using RIPPER, *in* '2014 Reliability and Maintainability Symposium', IEEE, pp. 1–7.

Gonzalez, E., Reder, M. & Melero, J. J. (2016), 'SCADA alarms processing for wind turbine component failure detection', *Journal of Physics: Conference Series* **753**(7).

Greenbyte AB (2017), 'Highlights from Greenbyte Forum 2017'. Accessed: 2018-09-11.
URL: *https://www.greenbyte.com/resources/blog/article/highlights-from-greenbyte-forum-2017/*

Guo, H., Watson, S., Tavner, P. & Xiang, J. (2009), 'Reliability analysis for wind turbines with incomplete failure data collected from after the date of initial installation', *Reliability Engineering and System Safety* **94**(6), 1057–1063.

Gupta, M. R., Bengio, S. & Weston, J. (2014), 'Training Highly Multiclass Classifiers', *Journal of Machine Learning Research* **15**, 1461–1492.

Hahn, B. (2017), Recommended Practice 17: WIND FARM DATA COLLECTION

AND RELIABILITY ASSESSMENT FOR O&M OPTIMIZATION, Technical Report May, IEA Wind.

Hahn, B., Durstewitz, M. & Rohrig, K. (2007), Reliability of Wind Turbines: Experiences of 15 years with 1,500 WTs, *in* 'Wind Energy: Proceedings of the Euromech Colloquium', number July, pp. 329–332.

Hameed, Z., Ahn, S. & Cho, Y. (2010), 'Practical aspects of a condition monitoring system for a wind turbine with emphasis on its design, system architecture, testing and installation', *Renewable Energy* **35**(5), 879–894.

Hastie, T., Tibshirani, R. & Friedman, J. (2009), *The Elements of Statistical Learning*, Vol. 1 of *Springer Series in Statistics*, Springer New York, New York, NY.

He, K., Zhang, X., Ren, S. & Sun, J. (2016), Deep Residual Learning for Image Recognition, *in* '2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)', pp. 770–778.

Hill, R. R., Peters, V. a., Stinebaugh, J. a. & Veers, P. S. (2009), 'Wind Turbine Reliability Database Update', *Wind Energy* (March).

Hoadley, K. A., Yau, C., Wolf, D. M., Cherniack, A. D., Tamborero, D., Ng, S., Leiserson, M. D., Niu, B., McLellan, M. D., Uzunangelov, V., Zhang, J., Kandoth, C., Akbani, R., Shen, H., Omberg, L., Chu, A., Margolin, A. A., Van't Veer, L. J., Lopez-Bigas, N., Laird, P. W., Raphael, B. J., Ding, L., Robertson, A. G., Byers, L. A., Mills, G. B., Weinstein, J. N., Van Waes, C., Chen, Z., Collisson, E. A., Benz, C. C., Perou, C. M., Stuart, J. M., Abbott, R., Abbott, S., Aksoy, B. A., Aldape, K., Ally, A., Amin, S., Anastassiou, D., Auman, J. T., Baggerly, K. A., Balasundaram, M., Balu, S., Baylin, S. B., Benz, S. C., Berman, B. P., Bernard, B., Bhatt, A. S., Birol, I., Black, A. D., Bodenheimer, T., Bootwalla, M. S., Bowen, J., Bressler, R., Bristow, C. A., Brooks, A. N., Broom, B., Buda, E., Burton, R., Butterfield, Y. S., Carlin, D., Carter, S. L., Casasent, T. D., Chang, K., Chanock, S., Chin, L., Cho, D. Y., Cho, J., Chuah, E., Chun, H. J. E., Cibulskis, K., Ciriello, G., Cleland, J., Cline, M., Craft, B., Creighton, C. J., Danilova, L., Davidsen, T., Davis, C., Dees, N. D., Delehaunty, K., Demchok, J. A., Dhalla, N., DiCara, D., Dinh, H., Dobson, J. R., Dodda, D., Doddapaneni, H. V., Donehower, L., Dooling, D. J., Dresdner, G., Drummond, J., Eakin, A., Edgerton, M., Eldred, J. M., Eley, G., Ellrott, K., Fan, C., Fei, S., Felau, I., Frazer, S., Freeman, S. S., Frick, J., Fronick, C. C., Fulton, L. L., Fulton, R., Gabriel, S. B., Gao, J., Gastier-Foster, J. M., Gehlenborg, N., George, M., Getz, G., Gibbs, R., Goldman, M., Gonzalez-Perez, A., Gross, B., Guin, R., Gunaratne, P., Hadjipanayis, A., Hamilton, M. P., Hamilton, S. R., Han, L., Han, Y., Harper, H. A., Haseley, P., Haussler, D., Hayes, D. N., Heiman, D. I., Helman, E., Helsel, C., Herbrich, S. M., Herman, J. G., Hinoue, T., Hirst, C., Hirst, M., Holt, R. A., Hoyle, A. P., Iype, L., Jacobsen, A., Jeffreys, S. R., Jensen, M. A., Jones, C. D., Jones, S. J., Ju, Z., Jung, J., Kahles, A., Kahn, A., Kalicki-Veizer, J., Kalra, D., Kanchi, K. L., Kane, D. W., Kim, H., Kim, J., Knijnenburg, T., Koboldt, D. C., Kovar, C., Kramer, R., Kreisberg, R., Kucherlapati, R., Ladanyi, M., Lander, E. S., Larson, D. E., Lawrence, M. S., Lee, D., Lee, E., Lee, S., Lee, W., Lehmann, K. V., Leinonen, K., Leraas, K. M., Lerner, S., Levine, D. A., Lewis, L., Ley, T. J., Li, H. I., Li, J., Li, W., Liang, H., Lichtenberg, T. M., Lin, J., Lin, L., Lin, P., Liu, W., Liu, Y., Liu, Y., Lorenzi, P. L., Lu, C., Lu, Y., Luquette, L. J., Ma, S., Magrini, V. J., Mahadeshwar, H. S., Mardis, E. R., Marra, M. A., Mayo, M., McAllister, C., McGuire, S. E., McMichael, J. F., Melott, J., Meng, S., Meyerson, M., Mieczkowski, P. A., Miller, C. A., Miller, M. L., Miller, M., Moore, R. A., Morgan, M., Morton, D., Mose, L. E., Mungall, A. J., Muzny, D., Nguyen, L., Noble, M. S., Noushmehr, H., O'Laughlin, M., Ojesina, A. I., Yang, T. H. O., Ozenberger, B., Pantazi, A., Parfenov, M., Park, P. J., Parker, J. S., Paull, E., Pedamallu, C. S., Pihl, T., Pohl, C., Pot, D., Protopopov, A., Przytycka, T., Radenbaugh, A., Ramirez, N. C., Ramirez, R., Rätsch, G., Reid, J., Ren, X., Reva, B., Reynolds, S. M., Rhie, S. K., Roach, J., Rovira, H., Ryan, M., Saksena, G., Salama, S., Sander, C., Santoso, N., Schein, J. E., Schmidt, H., Schultz, N., Schumacher, S. E., Seidman, J., Senbabaoglu, Y., Seth, S., Sharpe, S., Shen, R., Sheth, M., Shi, Y., Shmulevich, I., Silva, G. O., Simons, J. V., Sinha, R., Sipahimalani, P., Smith, S. M., Sofia, H. J., Sokolov, A., Soloway, M. G., Song, X., Sougnez, C., Spellman, P., Staudt, L., Stewart, C., Stojanov, P., Su, X., Sumer, S. O., Sun, Y., Swatloski, T., Tabak, B., Tam, A., Tan, D., Tang, J., Tarnuzzer, R., Taylor, B. S., Thiessen, N., Thorsson, V., Triche, T., Van Den Berg, D. J., Vandin, F., Varhol, R. J., Vaske, C. J., Veluvolu, U., Verhaak, R., Voet, D., Walker, J., Wallis, J. W., Waltman, P., Wan, Y., Wang, M., Wang, W., Wang, Z., Waring, S., Weinhold, N., Weisenberger, D. J., Wendl, M. C., Wheeler, D., Wilkerson, M. D., Wilson, R. K., Wise, L., Wong, A., Wu, C. J., Wu, C. C., Wu, H. T., Wu, J., Wylie, T., Xi, L., Xi, R., Xia, Z., Xu, A. W., Yang, D., Yang, L., Yang, L., Yang, Y., Yao, J., Yao, R., Ye, K., Yoshihara, K., Yuan, Y., Yung, A. K., Zack, T., Zeng, D., Zenklusen, J. C., Zhang, H., Zhang, J., Zhang, N., Zhang, Q., Zhang, W., Zhao, W., Zheng, S., Zhu, J., Zmuda, E. & Zou, L. (2014), 'Multiplatform analysis of 12 cancer types reveals molecular classification within and across tissues of origin', *Cell* **158**(4), 929–944.

Hu, R. L., Leahy, K., Konstantakopoulos, I. C., Auslander, D. M., Spanos, C. J. & Agogino, A. M. (2016), Using Domain Knowledge Features for Wind Turbine Diagnostics, *in* '2016 15th IEEE International Conference on Machine Learning and Applications (ICMLA)', IEEE, pp. 300–307.

Igba, J., Alemzadeh, K., Henningsen, K. & Durugbo, C. (2014), 'Effect of preventive maintenance intervals on reliability and maintenance costs of wind turbine gearboxes', *Wind Energy* **18**(11), 2013–2024.

International Electrotechnical Commission (IEC) (2010), 'IEC 61400-26-1: Time based availability for wind turbines'.

International Electrotechnical Commission (IEC) (2013), 'IEC 61400-12-1:2006: Wind Turbines - Part 12-1: Power Performance Measurements of Electricity Producing Wind Turbines'.

International Renewable Energy Agency (2018), Renewable Power Generation Costs in 2017, Technical report.

IPCC (2014), 'Summary for Policymakers'.

ISO (2016), 'ISO 14224:2016 Petroleum, petrochemical and natural gas industries – Collection and exchange of reliability and maintenance data for equipment'.

James, G., Witten, D., Hastie, T. & Tibshirani, R. (2013), *An Introduction to Statistical Learning*, Vol. 103 of *Springer Texts in Statistics*, Springer New York, New York, NY.

Kaidis, C., Uzunoglu, B. & Amoiralis, F. (2015), 'Wind turbine reliability estimation for different assemblies and failure severity categories', *IET Renewable Power Generation* **9**(8), 892–899.

Kandukuri, S. T., Klausen, A., Karimi, H. R. & Robbersmyr, K. G. (2016), 'A review of diagnostics and prognostics of low-speed machinery towards wind turbine farm-level health management', *Renewable and Sustainable Energy Reviews* **53**, 697–708.

Knerr, S., Personnaz, L. & Dreyfus, G. (1990), Single-layer learning revisited: a stepwise procedure for building and training a neural network, *in* 'Neurocomputing', number 68, Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 41–50.

Kothamasu, R., Huang, S. H. & VerDuin, W. H. (2006), 'System health monitoring and prognostics — a review of current paradigms and practices', *The International Journal of Advanced Manufacturing Technology* **28**(9-10), 1012–1024.

Kusiak, A. (2016), 'Renewables: Share data on wind energy', *Nature* **529**(7584), 19–21.

Kusiak, A. & Li, W. (2011), 'The prediction and diagnosis of wind turbine faults', *Renewable Energy* **36**(1), 16–23.

Kusiak, A. & Verma, A. (2011), 'A data-driven approach for monitoring blade pitch faults in wind turbines', *IEEE Transactions on Sustainable Energy* **2**(1), 87–96.

Kusiak, A. & Verma, A. (2012), 'Analyzing bearing faults in wind turbines: A data-mining approach', *Renewable Energy* **48**, 110–116.

Kusiak, A. & Verma, A. (2013), 'Monitoring wind farms with performance curves', *IEEE Transactions on Sustainable Energy* **4**(1), 192–199.

Kusiak, A., Zheng, H. & Song, Z. (2009), 'On-line monitoring of power curves', *Renewable Energy* **34**(6), 1487–1493.

Laouti, N., Sheibat-othman, N. & Othman, S. (2011), Support Vector Machines for Fault Detection in Wind Turbines, *in* '18th IFAC World Congress', pp. 7067–7072.

Lapira, E., Brisset, D., Davari Ardakani, H., Siegel, D. & Lee, J. (2012), 'Wind turbine performance assessment using multi-regime modeling approach', *Renewable Energy* **45**, 86–95.

Leahy, K., Bruton, K. & O' Sullivan, D. T. (2014), Implementing the Green Batch: A Case Study: Continuous Statistical Evaluation to Achieve the Most Energy Efficient and Reliable Process, *in* '19th IEEE International Conference on Emerging Technologies and Factory Automation'.

Leahy, K., Gallagher, C., Bruton, K., O'Donovan, P. & O'Sullivan, D. T. (2017), Automatically Identifying and Predicting Unplanned Wind Turbine Stoppages Using SCADA and Alarms System Data: Case Study and Results, *in* 'Journal of Physics: Conference Series', Vol. 926.

Leahy, K., Gallagher, C., O'Donovan, P., Bruton, K. & O'Sullivan, D. T. (2018), 'A Robust Prescriptive Framework and Performance Metric for Diagnosing and Predicting Wind Turbine Faults based on SCADA and Alarms Data with Case Study', *Energies* **11**(7), pp. 1–21.

Leahy, K., Gallagher, C., O'Donovan, P. & O'Sullivan, D. T. (2018), 'Cluster analysis of wind turbine alarms for characterising and classifying stoppages', *IET Renewable Power Generation* **12**(10), 1146–1154.

Leahy, K., Hu, R. L., Konstantakopoulos, I. C., Spanos, C. J. & Agogino, A. M. (2016), Diagnosing wind turbine faults using machine learning techniques applied to operational data, *in* '2016 IEEE International Conference on Prognostics and Health Management (ICPHM)', IEEE, pp. 1–8.

Leahy, K., Hu, R. L., Konstantakopoulos, I. C., Spanos, C. J., Agogino, A. M. & O'Sullivan, D. T. (2018), 'Diagnosing and Predicting Wind Turbine Faults from SCADA Data Using Support Vector Machines', *International Journal of Prognostics and Health Management* **9**(1), 1–11.

LEANWIND (2017), Driving Cost Reductions in Offshore Wind: The LEANWIND Project Final Publication, Technical Report 614020.

Lindahl, S. & Harman, K. (2012), Analytical techniques for performance monitoring of modern wind turbines, *in* 'European Wind Energy Association (EWEA) Conference', number April, pp. 1–9.

Link, H., Lacava, W., Dam, J. V. & Mcniff, B. (2011), Gearbox Reliability Collaborative Project Report : Findings from Phase 1 and Phase 2 Testing, Technical Report June, NREL.

Liu, X. Y., Wu, J. & Zhou, Z. H. (2006), Exploratory under-sampling for class-imbalance learning, *in* 'Proceedings - IEEE International Conference on Data Mining, ICDM', Vol. 39, pp. 965–969.

Lu, B., Li, Y., Wu, X. & Yang, Z. (2009), 'A review of recent advances in wind turbine condition monitoring and fault diagnosis', *Electronics and Machines in Wind* pp. 1–7.

Lydia, M., Kumar, S. S., Selvakumar, a. I. & Prem Kumar, G. E. (2014), 'A comprehensive review on wind turbine power curve modeling techniques', *Renewable and Sustainable Energy Reviews* **30**, 452–460.

Maples, B., Saur, G., Hand, M., van Pietermen, R. & Obdam, T. (2013), Installation, Operation, and Maintenance Strategies to Reduce the Cost of Offshore Wind Energy, Technical Report July, NREL.

Mc Garrigle, E. V., Deane, J. P. & Leahy, P. G. (2013), 'How much wind energy will be curtailed on the 2020 Irish power system?', *Renewable Energy* **55**, 544–553.

Mc Garrigle, E. V. & Leahy, P. G. (2015), 'Quantifying the value of improved wind energy forecasts in a pool-based electricity market', *Renewable Energy* **80**, 517–524.

Morice, C. P., Kennedy, J. J., Rayner, N. A. & Jones, P. D. (2012), 'Quantifying uncertainties in global and regional temperature change using an ensemble of observational estimates: The HadCRUT4 data set', *Journal of Geophysical Research Atmospheres* **117**(8).

Ngai, E. W., Hu, Y., Wong, Y. H., Chen, Y. & Sun, X. (2011), 'The application of data mining techniques in financial fraud detection: A classification framework and an academic review of literature', *Decision Support Systems* **50**(3), 559–569.

Nilsson, J. & Bertling, L. (2007), 'Maintenance management of wind power systems using condition monitoring systems—life cycle cost analysis for two case studies', *IEEE Transactions on Energy Conversion* **22**(1), 223–229.

Nowlan, F. S. & Heap, H. F. (1978), Reliability-centered maintenance, Technical report, United Air Lines Inc., San Francisco CA.

Noyes, J. (1999), *Alarm systems: a guide to design, management and procurement*, The Engineering and Materials Users Association.

O'Donovan, P., Gallagher, C., Leahy, K., Blake, S., Bruton, K. & O'Sullivan, D. T. (2017), A systematic mapping of industrial cyber-physical systems research for Industry 4.0, *in* 'International Manufacturing Conference'.

O'Donovan, P., Leahy, K., Bruton, K. & O'Sullivan, D. T. J. (2015*a*), 'An industrial big data pipeline for data-driven analytics maintenance applications in large-scale smart manufacturing facilities', *Journal of Big Data* **2**(1), 25.

O'Donovan, P., Leahy, K., Bruton, K. & O'Sullivan, D. T. J. (2015*b*), 'Big data in manufacturing: a systematic mapping study', *Journal of Big Data* **2**(1), 20.

Park, J.-y., Lee, J.-k., Oh, K.-y. & Lee, J.-s. (2014), 'Development of a Novel

Power Curve Monitoring Method for Wind Turbines and Its Field Tests', *IEEE Transactions on Energy Conversion* **29**(1), 119–128.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M. & Duchesnay, É. (2012), 'Scikit-learn: Machine Learning in Python', *The Journal of Machine Learning Research* **12**, 2825–2830.

Peter Tavner (2012), *Offshore Wind Turbines: Reliability, availability and maintenance*, Institution of Engineering and Technology.

Phua, C., Alahakoon, D. & Lee, V. (2004), 'Minority report in fraud detection', *ACM SIGKDD Explorations Newsletter* **6**(1), 50.

Qiu, Y., Feng, Y., Tavner, P., Richardson, P., Erdos, G. & Chen, B. (2012), 'Wind turbine SCADA alarm analysis for improving reliability', *Wind Energy* **15**(8), 951–966.

Reder, M. D., Gonzalez, E. & Melero, J. J. (2016), Wind Turbine Failures - Tackling current Problems in Failure Data Analysis, *in* 'Journal of Physics: Conference Series', Vol. 753.

Ribrant, J. & Bertling, L. M. (2007), 'Survey of failures in wind power systems with focus on Swedish wind power plants during 1997-2005', *IEEE Transactions on Energy Conversion* **22**(1), 167–173.

Richardson, P. (2010), Relating onshore wind turbine reliability to offshore application, PhD thesis, Durham University.

SAE (2009), 'Evaluation Criteria for Reliability-Centered Maintenance (Rcm) Processes'.

Saxena, A., Celaya, J., Balaban, E., Goebel, K., Saha, B., Saha, S. & Schwabacher, M. (2008), Metrics for evaluating performance of prognostic techniques, *in* '2008 International Conference on Prognostics and Health Management, PHM 2008'.

Schlechtingen, M. & Ferreira Santos, I. (2011), 'Comparative analysis of neural network and regression based condition monitoring approaches for wind turbine fault detection'.

Schlechtingen, M., Santos, I. F. & Achiche, S. (2013), 'Using data-mining approaches for wind turbine power curve monitoring: A comparative study', *IEEE Transactions on Sustainable Energy* **4**(3), 671–679.

Schmidt, G. A., Ruedy, R. A., Miller, R. L. & Lacis, A. A. (2010), 'Attribution of the present-day total greenhouse effect', *Journal of Geophysical Research Atmospheres* **115**(20), 1–6.

Sheng, S. (2012), Wind Turbine Gearbox Condition Monitoring Round Robin Study – Vibration Analysis, Technical Report July, National Renewable Energy Laboratory.

Sheng, S. & Veers, P. (2011), Wind Turbine Drivetrain Condition Monitoring - An Overview, *in* 'Machinery Failure Prevention Technology (MFPT): The Applied Systems Health Management Conference 2011', number October.

Skrimpas, G. A., Sweeney, C. W., Marhadi, K. S., Jensen, B. B., Mijatovic, N. & Holboll, J. (2015), 'Employment of Kernel Methods on Wind Turbine Power Performance Assessment', *IEEE Transactions on Sustainable Energy* **6**(3), 698–706.

Smil, V. (2016), *Energy Transitions: Global and National Perspectives*, 2 edn, ABC-CLIO.

Spinato, F. (2008), The Reliability Wind of Turbines, PhD thesis, Durham University.

Spinato, F., Tavner, P., van Bussel, G. & Koutoulakos, E. (2009), 'Reliability of wind turbine subassemblies', *IET Renewable Power Generation* **3**(4), 387.

Stehly, T., Heimiller, D. & Scott, G. (2016), 2016 Cost of Wind Energy Review, Technical Report December, NREL.

Stenberg, A. (2011), Wind energy statistics in Finland, *in* '67nd IEA Topical Expert Meeting - International Statistical Analysis on Wind Turbine Failures', Kassel, Germany, pp. 117–122.

Sustainable Energy Authority of Ireland (2017), Energy in ireland 1990-2016, Technical report.

Susto, G. A., Schirru, A., Pampuri, S., McLoone, S. & Beghi, A. (2015), 'Machine learning for predictive maintenance: A multiple classifier approach', *IEEE Transactions on Industrial Informatics* **11**(3), 812–820.

Tautz-Weinert, J. & Watson, S. J. (2017*a*), Challenges in Using Operational

Data for Reliable Wind Turbine Condition Monitoring, *in* ‘Proceedings of the Twenty-seventh (2017) International Offshore and Polar Engineering Conference (ISOPE)’, pp. 613–620.

Tautz-Weinert, J. & Watson, S. J. (2017*b*), Combining model-based monitoring and a physics of failure approach for wind turbine failure detection, *in* ‘30th Conference on Condition Monitoring and Diagnostic Engineering Management (COMADEM 2017)’, pp. 239–247.

Tautz-Weinert, J. & Watson, S. J. (2017*c*), ‘Using SCADA data for wind turbine condition monitoring – a review’, *IET Renewable Power Generation* **11**(4), 382–394.

Tchakoua, P., Wamkeue, R., Ouhrouche, M., Slaoui-Hasnaoui, F., Tameghe, T. A. & Ekemb, G. (2014), ‘Wind turbine condition monitoring: State-of-the-art review, new trends, and future challenges’, *Energies* **7**(4), 2595–2630.

Tomek, I. (1976), ‘Two Modifications of CNN’, *IEEE Transactions on Systems, Man and Cybernetics 6* pp. 769–772.

Uluyol, O., Parthasarathy, G., Foslien, W. & Kim, K. (2011), Power Curve Analytic for Wind Turbine Performance Monitoring and Prognostics, *in* ‘Annual Conference of the Prognostics and Health Management Society’, number August, pp. 1–8.

United Nations Framework Convention on Climate Change (2015), ‘Paris Agreement’.

U.S. Department of Energy Office of Energy Efficiency and Renewable Energy (2017), 2016 Wind Technologies Market Report, Technical report.

U.S. Energy Information Administration (2016), Capital Cost Estimates for Utility Scale Electricity Generating Plants, Technical Report November.

van Kuik, G. A. M., Peinke, J., Nijssen, R., Lekou, D., Mann, J., Sørensen, J. N., Ferreira, C., van Wingerden, J. W., Schlipf, D., Gebraad, P., Polinder, H., Abrahamsen, A., van Bussel, G. J. W., Sørensen, J. D., Tavner, P., Bottasso, C. L., Muskulus, M., Matha, D., Lindeboom, H. J., Degraer, S., Kramer, O., Lehnhoff, S., Sonnenschein, M., Sørensen, P. E., Künneke, R. W., Morthorst, P. E. & Skytte, K. (2016), ‘Long-term research challenges in wind energy – a research agenda by the European Academy of Wind Energy’, *Wind Energy Science* **1**(1), 1–39.

VGB PowerTech (2014), RDS-PP – Application Guideline; Part 32: Wind Power Plants, Technical report.

VGB PowerTech (2015), 'Levelised Cost of Electricity LCOE 2015', pp. 1–19.

Walford, C. A. (2006), Wind Turbine Reliability: Understanding and Minimizing Wind Turbine Operation and Maintenance Costs, Technical Report March, Albuquerque.

Wang, L., Zhang, Z., Xu, J. & Liu, R. (2016), 'Wind Turbine Blade Breakage Monitoring with Deep Autoencoders', *IEEE Transactions on Smart Grid* **3053**(c), 1–10.

Wennerhag, L. B. & Tjernberg, P. (2012), Wind Turbine Operation and Maintenance: Survey of the Development and Research Needs, Technical Report October, Elforsk.

Widodo, A. & Yang, B.-S. (2007), 'Support vector machine in machine condition monitoring and fault diagnosis', *Mechanical Systems and Signal Processing* **21**(6), 2560–2574.

Wilkinson, M., Harman, K., van Delft, T. & Darnell, B. (2014), 'Comparison of methods for wind turbine condition monitoring with SCADA data', *IET Renewable Power Generation* **8**(4), 390–397.

Wilkinson, M., Hendriks, B., Spinato, F. & Delft, T. V. (2011), Measuring Wind Turbine Reliability - Results of the Reliawind Project, *in* 'European Wind Energy Association Conference', pp. 1 – 8.

Wilson, D. L. (1972), 'Asymptotic Properties of Nearest Neighbor Rules Using Edited Data', *IEEE Transactions on Systems, Man and Cybernetics* **2**(3), 408–421.

WindEurope (2018), Wind in power 2017: Annual combined onshore and offshore wind energy statistics, Technical Report February.

Yang, W., Court, R. & Jiang, J. (2013), 'Wind turbine condition monitoring by the approach of SCADA data analysis', *Renewable Energy* **53**, 365–376.

Yang, W., Tavner, P. J., Crabtree, C. J., Feng, Y. & Qiu, Y. (2014), 'Wind turbine condition monitoring: Technical and commercial challenges', *Wind Energy* **17**(5), 673–693.

Yürüşen, N. Y., Tautz-Weinert, J., Watson, S. J. & Melero, J. J. (2017), The

Financial Benefits of Various Catastrophic Failure Prevention Strategies in a Wind Farm: Two market studies (UK-Spain), *in* 'Journal of Physics: Conference Series', Vol. 926.

Zhao, Y., Li, D., Dong, A., Kang, D., Lv, Q. & Shang, L. (2017), 'Fault prediction and diagnosis of wind turbine generators using SCADA data', *Energies* **10**(8), 1210.

Appendix A

Reder Taxonomy

Subsystem	Assembly
Power Module	
	Frequency Converter
	Generator
	Switch Gear
	Soft Starter
	MV/LV Transformer
	Power Feeder Cables
	Power Cabinet
	Power Module Other
	Power Protection Unit
Rotor & Blades	
	Pitch System
	Other Blade Brake
	Rotor
	Blades
	Hub
	Blade Bearing
Control & Communications	
	Sensors
	Controller
	Communication System
	Emergency Control & Communication Series
	Data Aquisition System
Nacelle	
	Yaw System
	Nacelle Cover
	Nacelle Bed plate
Drive Train	
	Gearbox
	Main Bearing
	Bearings
	Mechanical Brake
	High Speed Shaft
	Silent Blocks
	Low Speed (Main) Shaft

Subsystem	Assembly
Auxililiary System	
	Cooling system
	Electrical Protection & Safety
	Human Safety
	Hydraulic Group
	WTG Meteorological Station
	Lightning Protection
	Firefighting System
	Cabinets
	Service Crane
	Lift
	Grounding
	Beacon/Lights
	Power Supply Auxiliary Systems
	Electrical Auxiliary Cabling
Structure	
	Tower
	Foundations