Specification of 'normal' wind turbine operating behaviour for rapid anomaly detection: through the use of machine learning algorithms

Dissertation for the Degree of MSc in Renewable Energy Engineering
Heriot-Watt University

Nithiya M Streethran [1, *]

Supervisors: Dr Nick Bennett [2] and Dr Iain Dinwoodie [3]

[1, 2] School of Engineering and Physical Sciences, Heriot-Watt University, Edinburgh EH14 4AS, United Kingdom
[3] Natural Power Consultants, Ltd., Stirling FK7 7XE, United Kingdom

* Email: nmstreethran@gmail.com

Submitted on 25 August, 2017

Reformatted and recompiled on December 17, 2019

Abstract

Maximising the economic effectiveness of a wind farm is essential in making wind a more economic source of energy. This effectiveness can be increased through the reduction of operation and maintenance costs, which can be achieved through continuously monitoring the condition of wind turbines. An alternative to expensive condition monitoring systems, which can be uneconomical especially for older wind turbines, is to implement classification algorithms on supervisory control and data acquisition (SCADA) signals, which are collected in most wind turbines. Several publications were reviewed, which were all found to use separate algorithms to predict specific faults in advance. In reality, wind turbines tend to have multiple faults which may happen simultaneously and have correlations with one another. This project focusses on developing a methodology to predict multiple wind turbine faults in advance simultaneously by implementing classification algorithms on SCADA signals for a wind farm with 25 turbines rated at 2,500 kW, spanning a period of 30 months. The data, which included measurements of wind speed, active power and pitch angle, was labelled using corresponding downtime data to detect normal behaviour, faults and varying timescales before a fault occurs. Three different classification algorithms, namely decision trees, random forests and k nearest neighbours were tested using imbalanced and balanced training data, initially to optimise a number of hyperparameters. The random forest classifier produced the best results. Upon conducting a more detailed analysis on the performance of specific faults, it was found that the classifier was unable to detect the varying timescales before a fault with accuracy comparable to that of normal or faulty behaviour. This could have been due to the SCADA data, which are used as features, being unsuitable for detecting the faults, and there is potential to improve this by balancing only these classes.

Keywords: wind turbine, classification algorithm, SCADA, fault detection, condition monitoring

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

There is a need to increase the economic effectiveness of wind turbines, which refers to the cost to run them relative to the electricity generation, or revenue [1], [2]. Increasing this effectiveness lowers the payback period of new wind turbines or farms, thus making wind a more economic clean energy source, attracting governments and private organisations to make more investments in wind projects [1]. It can, however, be decreased due to major component failure, frequent downtime, turbine degradation and age, which in turn increase the operation and maintenance cost and decrease the energy generation efficiency of wind turbines [1], [3]. There are difficulties and high costs involved in carrying out maintenance on wind turbines, especially for ones that operate in extreme and remote conditions, such as offshore wind farms, where the turbines tend to also exist in larger numbers [3], [4].

Condition-based monitoring systems that continuously monitor wind turbine states increase this effectiveness by significantly reducing the maintenance costs, reportedly by 20 % to 25%, as it prevents unscheduled maintenance [2]. According to the Electric Power Research Institute, reactive maintenance, which refers to running the turbine until it reaches failure, has the highest cost, followed by preventive or scheduled maintenance, which is reported to cost 24 % less [5]. Meanwhile, condition-based or predictive maintenance, which prevents catastrophic failure, [1] is reported to save 47 % of the cost of reactive maintenance, [5] which makes it the most cost-effective and preferred approach. Condition-based monitoring technologies include sensor-based oil and vibration analysis, which are useful for checking the oil for properties such as temperature, and rotating equipment respectively [6]. These technologies, however, tend to put emphasis on the more expensive parts of a wind turbine such as the gearbox [7] due to the high costs involved in the installation of these sensors [2], [6]. These systems, which can be purchased from the turbine manufacturer, are usually pre-installed in offshore wind turbines due to the harsh environments in which they operate. However, they can be expensive [4] and uneconomical, especially for older wind turbines in onshore wind farms, whose outputs are often less than that of an offshore wind farm.

An alternative would be to use SCADA-based analysis, where the only cost involved would be computational and expensive sensors are not required [2], [4]. A SCADA system, which stands for supervisory control and data acquisition, found pre-installed in most utility-scale wind turbines, collects data using numerous sensors at the controllers with usually 10-minute resolution [4], [8], of various parameters of the wind turbine, such as wind speed, active power, bearing temperature and voltage [2]. Power curve analysis can be done using this data, but this analysis only detects wind turbine underperformance [9]. Meanwhile, implementing machine learning algorithms on SCADA signals to classify them as having either normal or anomalous behaviour, has the ability to predict faults in advance. This has been demonstrated in a number of publications.

Kusiak and Li [10] investigated predicting a specific fault, which is diverter malfunction. 3 months' worth of SCADA data of four wind turbines were used and the corresponding status and fault codes were integrated into this data to be labelled to differentiate between normal and fault points. To prevent bias in prediction in machine learning, the labelled data is sampled at random, ensuring the number of samples with a fault code is comparable to the number of normal samples. Four classification algorithms, namely neural networks, boosting

tree, support vector machines, and classification and regression trees were trained using two-thirds of this data which was randomly selected. The boosting tree, which was found to have the highest accuracy of 70 % for predicting specific faults, was investigated further. The accuracy of predicting a specific fault at the time of fault was 70 %, which decreased to 49 % for predicting it 1 hour in advance. Only one specific fault was the focus of this methodology and in reality, wind turbines could have many faults in different components and structures, which may all have some form of correlation between one other.

Godwin and Matthews [7] focussed on wind turbine pitch control faults using a classifier called the RIP-PER algorithm. They used 28 months' worth of SCADA data containing wind speeds, pitch motor torques and pitch angles, of eight wind turbines known to have had pitch problems in the past. The classes used were normal, potential fault and recognised fault. Using maintenance logs, data up to 48 hours in advance was classed as recognised fault, data in advance of this with corresponding SCADA alarm logs indicating pitch problems was classed as potential fault, and the remaining unclassed data was classed as normal. Random sampling was performed here as well to balance the classes and prevent bias. The data of four turbines were used to train the RIPPER algorithm, and the remaining four used for testing. The analysis was done using the entire 28 months of data as well as 24, 20, 16, 12, 8 and 4 months of data to find out how the amount of data affects the accuracy of classification. Using the entire 28 months of data was found to produce the most accurate classifier, with a mean accuracy of 85 %. Looking at the results in more depth, it was found that the classifier had F1 scores, which is an accuracy measure that accounts for true and false positives and negatives, of 79 %, 100 % and 78 % in classifying normal, potential fault and recognised fault data respectively. Although the results are an improvement to Kusiak and Li [10], this methodology similarly focussed on only one fault.

Leahy et al. [2] used a specific fault prediction approach, implementing a support vector machine classifier from scikit-learn's LibSVM. They used SCADA data from a single 3 MW wind turbine spanning 11 months with status and warning codes. The labelling was done such that data with codes corresponding to the turbine in operation, low and storm wind speeds represent normal conditions and codes corresponding to each specific fault to represent faulty conditions. Data preceding these fault points by 10 minutes and 60 minutes were also labelled as faults in separate sets and the effects of using these different time scales to predict faults were investigated. For data identified as normal, filters were applied to remove curtailment and anomalous points. The classifier's hyperparameters were optimised using randomised grid search and validated using ten-fold cross-validation, and the classes were balanced using class weights. Separate binary classifiers were trained to detect each specific type of fault, which were faults in air cooling, excitation, generator heating, feeding and mains failure. The prediction of generator heating faults 10 minutes in advance had the best results, with F1 scores of 71 % and 100 % using balanced and imbalanced training data respectively. This increase in score using imbalanced data was attributed to the test set having very few instances with the fault class relative to normal data. The same fault, when predicted 60 minutes in advance, had F1 scores of 17 % and 100 % using imbalanced and balanced training data respectively. Although the score is perfect and it demonstrates the effects of using balanced datasets, the classification again is done separately for each specific fault and it performed poorly on other faults. For instance, detecting excitation faults 10 and 60 minutes in advance using balanced training data only yielded F1 scores of 8 % and 27 % respectively.

This project will therefore focus on integrating the ability to predict multiple faults at different time scales simultaneously.

1.2 Objectives

The first objective of this project is to implement a classification algorithm on wind turbine SCADA signals to identify underperforming turbines. This involves setting-up the machine learning environment, processing operational data and reporting initial results obtained through implementing a classification algorithm on the data.

The second objective is to create an effective methodology for the integration of failures and to present and interpret results. This includes labelling the data such that each specific fault can be differentiated, evaluating the performances of several classification algorithms to find the most suitable classifier, identifying limitations and suggesting improvements to the method and how it can be adapted for use in industry.

1.3 Outline

Chapter 2 will describe in detail the tools and datasets used, how the data was processed and labelled and the classification methods and performance metrics used. In Chapter 3, a detailed description of the results obtained is presented, followed by a discussion of these results and limitations of this methodology in Chapter 4. In Chapter 5, conclusions are drawn and possible areas for future work are recommended.

Chapter 2

Methodology

2.1 Tools and datasets

This project requires a computer with Python Programming Language [11] and essential libraries installed for data processing. The computer used has a dual core processor with 2.8 GHz maximum clock speed and 4 GB RAM. Additionally, the open source scikit-learn library [12] is used for machine learning. The datasets used are that of a wind farm comprised of 25 turbines with a rated power of 2,500 kW covering a period of 30 months starting 1st November 2014, downloaded from Natural Power's database in CSV format. The first dataset is wind turbine SCADA signals timestamped with a resolution of 10 minutes, with a total file size of 452 MB, and the other dataset is corresponding downtime data for the same period, with a total file size of 4 MB. In the interests of Natural Power, the location of the wind farm and turbine model will not be disclosed in this report.

2.2 Data processing

The SCADA data has 17 fields, summarised in Table 2.1. Fields highlighted in green are average measurements recorded over each 10-minute period. Since these highlighted fields are properties of the turbines or describe its performance, they can be used as features in machine learning. Each turbine has two nacelle anemometers and wind vanes; one is used to control the turbine, and the other to monitor the first. The measurements from the anemometer and wind vane used to control the turbine are recorded again as `ws_av` and `wd_av`, with the latter taking into account the nacelle position. Using only `ws_av` and `wd_av` for wind speed and wind direction, the number of features that are available for machine learning is 10.

The downtime data consists of fields summarised in Table 2.2. Each row of downtime data consists of the start and end timestamps of the downtime event, downtime categories, workorders and alarms. Downtime categories, which are turbine, environmental, grid, infrastructure and availability categories, describes the turbine's condition or cause of downtime when the maintenance work was undertaken. Each condition within each downtime category is represented by a unique identifier in the dataset. A separate spreadsheet accompanying the dataset list what each identifier stands for. All quantities in the downtime data, except the alarms, are supervised (i.e., the data recordings are input and monitored by maintenance professionals).

Each row of SCADA data requires a class which describes the state of the turbine. The chosen classes are 'normal' for normal behaviour, and 'faulty' to signify a fault. As the aim is to predict faults in advance, a category of classes, called 'before fault' will also be used. To automate the labelling process, the SCADA data can be merged with the downtime data, which has turbine categories, listed in Table 2.3, that can be used to label faults. Some of these turbine categories, such as 'OK' and 'scheduled maintenance', do not indicate a fault in the turbine, and 'other' does not specify the condition. Therefore, only the turbine categories which indicate faults, highlighted in green, are used to class the SCADA data. Prior to merging the two datasets, the downtime data is restructured such that it has the same 10-minute resolution as the SCADA data. The SCADA data was also found to have missing rows of data. Empty data rows with only the timestamp corresponding to the missing rows were added to rectify this. Once they are merged, 14 separate labels, or columns, are added for each specific fault, which will allow for the different faults to be distinguished. The rows with a fault category

Table 2.1: Summary of SCADA fields for the SCADA data used in this project. The fields include timestamps with a resolution of 10 minutes, average active power, wind speed, pitch and runtime. The fields that contain measurements averaged over the 10-minute period are highlighted in green. These measurements can be used as features in machine learning as they are turbine properties.

SCADA field	Description	Unit
timestamp	In the format dd/mm/YYYY HH:MM:SS, every 10 minutes	
turbine_id	Turbine identifier (1 to 25)	
ap_av	Average active power	kW
ap_dev	Active power deviation	kW
ap_max	Maximum active power	kW
reactive_power	Reactive power	kVAr
ws_1	Wind speed measured by nacelle anemometer 1	m/s
ws_2	Wind speed measured by nacelle anemometer 2	m/s
ws_av	Anemometer wind speed (either ws_1 or ws_2)	m/s
wd_1	Wind direction measured by wind vane 1	°
wd_2	Wind direction measured by wind vane 2	°
gen_sp	Generator speed	rpm
rs_av	Rotor shaft speed	rpm
nac_pos	Nacelle position	°
wd_av	Corrected wind direction (nac_pos + (either wd_1 or wd_2))	°
pitch	Pitch angle	°
runtime	Number of seconds the turbine has operated in the 10-minute period	s

Table 2.2: Summary of fields for the downtime data used in this project. The fields include start and end timestamps for the downtime event, downtime categories, workorders and alarms.

Downtime field	Description
timestamp_start	Start time of event, in the format dd/mm/YYYY HH:MM:SS
timestamp_end	End time of event, in the format dd/mm/YYYY HH:MM:SS
turbine_id	Turbine identifier (1 to 25)
alarm_id	Ranging from 1 to 480, each corresponding to a turbine status
GridCategory_id	Identifier (0 to 3); describes the grid status (e.g., planned outage, unplanned outage, ...)
InfrastructureCategory_id	Identifier (0 to 3); describes the infrastructure status (e.g., planned outage, unplanned outage, ...)
EnvironmentalCategory_id	Identifier (0 to 14); describes the condition of the operating environment (e.g., icing, turbulence, ...)
TurbineCategory_id	Identifier (0 to 22); describes the turbine's condition or problem (e.g., yaw system, electrical controls, ...)
AvailabilityCategory_id	Identifier (0 to 2); describes the availability status (e.g., available, not available)
comment	Elaborates the condition or maintenance work undertaken
workorder_id	Recorded when maintenance work is undertaken

Table 2.3: List of turbine categories in the wind farm downtime data. The categories used as the different faults for labelling are highlighted in green. The others do not indicate a fault.

Turbine category											
id	Name		id	Name		id	Name		id	Name	
0	Unknown		6	Generator		12	Unlogged manual stop		18	Cable unwind	
1	OK		7	Yaw system		13	Customer stop		19	Hub	
2	Anemometry		8	Electrical controls		14	Noise constraints		20	Rotor blades	
3	Rotor brake		9	Hydraulics		15	Scheduled maintenance		21	Delayed startup	
4	Main shaft		10	Electrical system		16	Tower		22	Other	
5	Gearbox		11	Pitch control		17	Retrof t				

are classed as 'faulty' in the corresponding column.

To summarise the machine learning terminology used, features refer to SCADA fields which are turbine properties, labels refer to turbine categories or type of fault, and classes refer to the state of the turbine (e.g., 'normal' or 'faulty') for each row of data at each label. The features and labels will be fit to a classifier for training as arrays X of size [rows, 10] and Y of size [rows, 14] respectively, where rows refer to the number of rows in the training data.

To predict faults for each label, rows with timestamps up to 48 hours in advance of a 'faulty' row are classed at 6-hour intervals (i.e., up to X hours before a fault, where X = 6, 12, ..., 48). The reasons for having classes of 6-hour intervals for fault detection rather than a single class is to allow action to be taken appropriate to the time before fault. For example, if it is predicted that the wind turbine could have a fault in six hours or less, it could be switched off to prevent further damage from occurring. 48 hours is enough time for maintenance professionals to travel to site and carry out inspection, and decide on what action to take. Depending on the nature of the site, this value can be modified (i.e., for an offshore wind farm which operates in harsh environments, it is more likely to take a longer time to travel to the site and complete works relative to an onshore wind farm).

2.3 Classification

Chapter 3

Results

Chapter 4

Discussion

Chapter 5

Conclusion

References

[1]   K. Kim, G. Parthasarathy, O. Uluyol, W. Foslien, S. Sheng, and P. Fleming, "Use of SCADA Data for Fail-ure Detection in Wind Turbines," presented at the ASME 2011 5th International Conference on Energy Sustainability, American Society of Mechanical Engineers Digital Collection, Mar. 13, 2012, pp. 2071–2079, DOI: 10.1115/ES2011-54243.

[2]   K. Leahy, R. L. Hu, I. C. Konstantakopoulos, C. J. Spanos, and A. M. Agogino, "Diagnosing wind tur-bine faults using machine learning techniques applied to operational data," in *2016 IEEE International Conference on Prognostics and Health Management (ICPHM)*, Jun. 2016, pp. 1–8, DOI: 10.1109/ICPHM.2016.7542860.

[3]   S. Dienst and J. Beseler, "Automatic Anomaly Detection in Offshore Wind SCADA Data," 2016.

[4]   J. Tautz-Weinert and S. J. Watson, "Using SCADA data for wind turbine condition monitoring – a re-view," *IET Renewable Power Generation*, vol. 11, no. 4, pp. 382–394, 2017, ISSN: 1752-1424, DOI: 10.1049/iet-rpg.2016.0248.

[5]   "Wind Turbine Condition Monitoring," National Instruments, White Paper, Dec. 18, 15, [Online]. Available: http://www.ni.com/white-paper/9231/en/ (visited on Aug. 19, 2017).

[6]   F. P. García Márquez, A. M. Tobias, J. M. Pinar Pérez, and M. Papaelias, "Condition monitoring of wind turbines: Techniques and methods," *Renewable Energy*, vol. 46, pp. 169–178, Oct. 1, 2012, ISSN: 0960-1481, DOI: 10.1016/j.renene.2012.03.003.

[7]   J. L. Godwin and P. Matthews, "Classification and Detection of Wind Turbine Pitch Faults Through SCADA Data Analysis," *International Journal of Prognostics and Health Management*, vol. 4, 2013.

[8]   W. Yang, P. J. Tavner, C. J. Crabtree, Y. Feng, and Y. Qiu, "Wind turbine condition monitoring: Technical and commercial challenges," *Wind Energy*, vol. 17, no. 5, pp. 673–693, 2014, ISSN: 1099-1824, DOI: 10.1002/we.1508.

[9]   S. Gill, B. Stephen, and S. Galloway, "Wind turbine condition assessment through power curve copula modeling," *IEEE Transactions on Sustainable Energy*, vol. 3, pp. 94–101, 2012, DOI: 10.1109/TSTE.2011.2167164.

[10] A. Kusiak and W. Li, "The prediction and diagnosis of wind turbine faults," *Renewable Energy*, vol. 36, no. 1, pp. 16–23, Jan. 1, 2011, ISSN: 0960-1481, DOI: 10.1016/j.renene.2010.05.014.

[11] Welcome to Python.org, [Online]. Available: https://www.python.org/ (visited on May 1, 2018).

[12] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, and M. Perrot, "Scikit-learn: Machine Learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011, [Online]. Available: http://jmlr.org/papers/volume12/pedregosa11a/pedregosa11a.pdf.