Automatically Identifying and Predicting Unplanned Wind Turbine Stoppages Using SCADA and Alarms System Data: Case Study and Results

Automatically Identifying and Predicting Unplanned Wind Turbine Stoppages Using SCADA and Alarms System Data: Case Study and Results

Kevin Leahy[1], **Colm Gallagher**[1], **Ken Bruton**[2], **Peter O'Donovan**[1] and **Dominic T.J. O'Sullivan**[1]

[1]Department of Civil & Environmental Engineering, University College Cork, Ireland
[2]Department of Mechanical, Biomedical and Manufacturing Engineering, Cork Institute of Technology, Cork, Ireland

E-mail: `kevin.leahy@umail.ucc.ie`

Abstract. Using 10-minute wind turbine SCADA data for fault prediction offers an attractive way of gaining additional prognostic capabilities without needing to invest in extra hardware. To use these data-driven methods effectively, the historical SCADA data must be labelled with the periods when the turbine was in faulty operation as well the sub-system the fault was attributed to. Manually identifying faults using maintenance logs can be effective, but is also highly time consuming and tedious due to the disparate nature of these logs across manufacturers, operators and even individual maintenance events. Turbine alarm systems can help to identify these periods, but the sheer volume of alarms and false positives generated makes analysing them on an individual basis ineffective. In this work, we present a new method for automatically identifying historical stoppages on the turbine using SCADA and alarms data. Each stoppage is associated with either a fault in one of the turbine's sub-systems, a routine maintenance activity, a grid-related event or a number of other categories. This is then checked against maintenance logs for accuracy and the labelled data fed into a classifier for predicting when these stoppages will occur. Results show that the automated labelling process correctly identifies each type of stoppage, and can be effectively used for SCADA-based prediction of turbine faults.

1. Introduction

Due to the nature of the dynamic conditions in which they operate, operations and maintenance (O&M) costs for wind turbines can account for up to 30% of the cost of generation of energy [1, 2, 3]. In the case of offshore turbines, access for routine maintenance can be expensive and weather dependent, with the costs for unscheduled repairs or inspections being even higher. Transport costs for a maintenance crew to an offshore turbine ranges from €85/hr on relatively small vessels, to over €400/hr for larger vessels and helicopters which are not as dependent on calm sea conditions [4]. Because of this, it is important to have robust control and monitoring strategies which can detect faults and notify technicians without raising false alarms.

If a specific type of fault, along with an estimate for the time to failure, can be predicted in advance, preventative measures can be taken to either avoid the fault or schedule maintenance at the optimal time. For example, scheduling maintenance during good weather for offshore turbines, or during other planned maintenance activities for maximum economic benefit [4]. As well as this, information about any predicted turbine downtime that may be unavoidable is very useful to grid operators. 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. In Ireland, the new Integrated Single Electricity Market will have this requirement from 2018 [5].

Fault prognostic functions have traditionally been performed by condition monitoring systems (CMSs) based on vibration or oil particulate sensors or similar. These can cost upwards of €13,000 per turbine and, due to the inherent wide range of operating conditions associated with wind energy, have not performed as well as CMSs in other industries [6, 7]. CMSs leveraging pre-existing low-frequency (generally 10-minute period) SCADA data can avoid the high installation costs associated with traditional CMSs [8]. Using this data to to detect, diagnose and predict faults and turbine downtime has seen some success in the past. However, problems with data availability and some ambiguity in how faults are defined have complicated efforts.

1.1. Related Works

Evidence of performance degradation leading up to large or catastrophic faults has been well documented in the literature. In [9], the authors showed that normal behaviour models showed a residual that indicated a fault was imminent leading up to major maintenance works. The authors in [10] also use a residual model for prognostics on the main bearing of a wind turbine. Their work shows that some indication of a fault is possible more than 30 days in advance of failure, but that further testing is needed to verify. The work done in [11] once again uses normal behaviour modelling, this time to detect anomalous temperatures indicating generator and gearbox faults. The results show that some indications of faults are possible in advance, but again more work is needed to prove effectiveness.

Other work has shown varying degrees of success in predicting less serious, but more frequent faults, with shorter prognostic horizons. These methods generally use a classification-based approach, where SCADA samples are classified as "possibly faulty" or "fault-free" in advance of a fault occurring. The authors in [12] found that prediction of a specific type of fault was possible with 67% accuracy and 73% recall 30 minutes in advance of the fault occurring. However, here, the test set used for evaluation had undersampled the fault-free data and did not represent a true distribution. In [13], the authors use the RIPPER decision tree algorithm to derive human-readable rules for giving an indication of pitch fault occurrence, with precision and recall scores of 0.8 and 0.75, respectively, on a 48 hour prognostic horizon. Once again, however, these scores were based on a validation set which undersampled the fault-free class. When tested against a full set of data, recall scores were still around 87%, but precision was reduced to 25%. However, there was a 52% reduction in the number of false alarms when compared to the SCADA alarm system, lowering the burden of analysis for maintenance operators. The prediction of specific blade pitch faults was demonstrated in [14] using genetically programmed decision trees. Here, the maximum prediction time was 10 minutes, at a 68% accuracy and 71% recall. However, the SCADA data used was at a resolution of 1s rather than the more common 10 minutes. Previous work by the authors in [15] showed that prediction of generator heating faults was possible one hour in advance with a recall score of >0.9. This performance was brought down by poor precision of 0.56, meaning there were too many false alarms for it to be field deployable, though this score was significantly improved with the construction of additional features in [16].

1.2. Defining a Fault

In all of the previous work mentioned, a significant point to keep in mind is how "faults" are defined. In some cases, such as [12, 15], periods when turbine "fault alarms" were active are used to indicate the presence of a fault. These are alarms generated by the turbine SCADA system and are generally triggered when the controller detects an operating condition outside of acceptable bounds, e.g. over-speed, over-temperature or a control issue. Hence, single instances of turbine alarms are not always indicative of a fault having taken place, and can often be false alarms [17]. In other cases, faults are defined as being the result of a maintenance action taking place, as depicted in the maintenance logs [13]. This avoids the issue of clouding the data with false alarms and very brief stoppages intended to reset the control system rather than as the result of damage or to avoid damage taking place. However, labelling data using maintenance logs can be a tedious process, as logs are stored as written documents rather than in dedicated databases [18]. As well as this, the logs are not always standardised across different OEMs or operators, or, in some cases, even different maintenance events, meaning automating this process by using OCR text recognition or reading spreadsheets can be very involved.

One way to get the best of both worlds is by identifying periods of downtime and only labelling periods where downtime was significant (e.g. >12 hours) as being indicative of a fault taking place. Although this avoids the problems associated with false alarms, it means that some shorter term stoppages are not predicted, and the root cause or even sub-system which the fault belongs to are not immediately clear.

In this paper we present a method for identifying periods of faulty operation in the SCADA data without needing to manually reference maintenance logs by assigning different rules to generated alarms, and use this to perform fault prediction using classification techniques. In section 2, we give an overview of the data used in this study. In section 3, we describe the automated fault and stoppage labelling process and evaluate its accuracy against maintenance logs and availability data. In section 4, the classification techniques used to predict when these faults occur are discussed. In section 5, the results of this fault prediction are discussed, while conclusions are drawn and future work described in section 6.

2. Description of Data

The data used in this study composes of 6 months of data from 11 2.5MW DFIG wind turbines in the East of Ireland. The dataset comprises of three parts: SCADA data, alarms data and availability data. Additionally, maintenance logs, which consist of spreadsheet and PDF documents, are used to validate the automated fault-identification process.

2.1. SCADA Data

The turbine control system monitors many instantaneous variables such as power output, wind speed, temperatures of various components, blade angles, etc. These are aggregated into 10 minute averages, and in some cases the maximum, minimum and standard deviation of values that occurred in each 10 minute period are also included. In this case, there were over 300,000 samples spread across all turbines, with each sample having over 90 different variables. A sample of this data is seen in table 1.

Table 1. Ten Minute SCADA Data

TimeStamp	Wind Speed (avg.) m/s	Wind Speed (std.) m/s	Power (avg.) kW	Gen. RPM (avg.) rpm	Bearing Temp (avg.) °C
09/06/2015 14:10:00	5.8	0.1	367	756	25
09/06/2015 14:20:00	5.7	0.1	378	750	25
09/06/2015 14:30:00	5.6	0.5	384	747	25
09/06/2015 14:40:00	2.8	0.9	0	0	24

Table 2. Random Sample of Alarms Data from Turbine 2

t_s	t_e	Code	Description	Category	Severity
2015/06/01 02:13:38	17:25:05	a_{41}	Normal Operation	No Fault	Information
2015/06/02 10:19:05	10:19:05	a_{124}	Motor Fuse	Generator	Warning
2015/06/03 07:56:14	07:57:32	a_{91}	Low wind cut out	Weather	Information
2015/11/04 08:38:27	08:38:37	a_{81}	Line CCU current	Freq. Conv.	Fault
2016/02/01 01:26:46	01:26:54	a_{22}	Normal Operation	No Fault	Information
2016/02/02 15:05:38	15:11:01	a_{51}	Emergency stop	User	Critical Fault

2.2. Alarms Data
Turbine alarm systems vary between manufacturers, but generally behave in similar ways. At the most basic level, a turbine alarm (also called status message or event by some manufacturers) is generated to indicate that the turbine's operating state has changed. OEMs generally attribute at least three different levels of severity to turbine alarms

- *Information alarms* are generally to communicate changes in certain operating conditions, e.g. when the wind speed is too low for generation, 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 come close to exceeding certain thresholds.
- *Fault alarms* are generated when these thresholds are exceeded.

Fault alarms generally cause the turbine to shut down. Depending on the severity of the fault alarm itself, either an automatic remote reset, a manual remote reset, or a manual on-site inspection is required to reset the alarm. Alarms which require on-site inspections are generally alarms which do not often give false positives, i.e. they are usually related to safety critical components and only fire if these components are damaged or not operating correctly (e.g. the emergency braking system). In general, however, alarms which only need a remote reset to clear can often generate a lot of false positives. Due to the sheer volume of individual alarms generated, they can be overlooked by maintenance operators [17]. A sample of alarms data used in this study can be seen in table 2.

Altogether, there were over 100,000 alarm instances in the dataset. As can be seen, each instance has an associated start time, t_s, end time, t_e, code (here made anonymous for identity purposes), description, category and type. The category relates to its functional location on the turbine, while the severity refers to whether the alarm is an information, warning or fault alarm. Note that fault alarms for this particular manufacturer are also split into "fault" and "critical fault" categories, with "critical fault" alarms requiring a manual on-site reset to restart the turbine.

2.3. Availability Data
The turbine availability data tracks the availability states of the turbine over every 10 minute period in the SCADA data. These are essentially counters which document how many seconds in every 10 minute period the turbine was in one of the following states:

- SOT (System OK) - the turbine was operating normally
- DT (Down Time) - the turbine was down due to a fault on the turbine
- LOT (Line Out Time) - the turbine was not operating due to a grid fault
- WOT (Weather Out Time) - the turbine was not operating, or was curtailed, because of severe weather

- MT (Maintenance Time) - the turbine was down for routine maintenance
- RT (Repair Time) - the turbine was down for unplanned repairs

A sample of this data can be seen in table 3.

Table 3. Ten Minute Availability Data

TimeStamp	SOT	DT	LOT	WOT	MT	RT
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	
09/06/2015 14:40:00	100	500	0	0	0	0
09/06/2015 14:50:00	600	0	0	0	0	0

3. Labelling Faults

As mentioned previously, labelling the SCADA data for fault classification can be a tedious step. Using maintenance logs involves a lot of manual intervention, while simply labelling the data according to times when alarms were active can mean that a lot of false positives are labelled as faults. As well as this, generally during a turbine stoppage related to a fault, there can be a number of alarms generated related to different subsystems, so it is not always clear which subsystem the original fault is attributed to. In this work, we outline a novel method of identifying the periods of faulty operation by cross referencing with particular sequences of alarms.

First, alarms which cause the turbine to shutdown, or only appear as a result of the turbine shutting down, are identified. We name this set of alarms A_s. These alarms are then assigned one of the following stop categories:

- Fault categories - These are alarms which cause the turbine to shut down due to a fault on the turbine. These are labelled according to the subsystem they belong to (note that these can differ from the OEM-assigned category seen in table 2):
  - *fc* - frequency converter faults
  - *pt* - pitch faults
  - *gb* - gearbox faults
  - *gn* - generator faults
  - *sn* - sensor faults
  - *to* - tower faults
  - *az* - azimuth/rotor faults
  - *bk* - brake faults
- *no* (normal operation) - the alarm causes the turbine to shut down, but due to things that are considered part of normal operation, e.g. cable untwisting, or running system tests.
- *wa* (weather) - the turbine is down due to severe wind conditions
- *ma* (manual/maintenance) - the alarm causes the turbine to shut down as a result of manual intervention, e.g. maintenance activities or a remote manual shut down.
- *gd* (grid) - these alarms notify the operator that the turbine has been shut down as a result of a grid fault, i.e. a fault not caused by the turbine

3.1. Alarm Batches

Any time the turbine shuts down, a number of alarms are generated. These alarms may persist and be active for the duration of the shutdown, or may be cleared before the turbine comes back online. The gap between the alarm being "cleared" (usually as a result of a remote reset) and the turbine coming back online and generating, however, can be a number of hours if the turbine needs a site visit for manual inspection. As well as this, many alarms may be triggered concurrently during a stoppage, meaning that it is difficult to attribute the stoppage to any single alarm. This means that if trying to identify periods of faulty operation to label SCADA data for fault prediction, labelling data according to specific instances of alarms and the times they were present can lead to erroneous classification.

To avoid this problem, we instead look at *batches* of alarm sequences that appear during individual turbine stoppages. The first step in creating these batches from the data is by identifying the alarm code that signifies the turbine returning to normal operation after a stop, referred to here as a_n. Each batch is then created for every turbine as follows (note that "A_s alarm" here refers to an alarm with a code from the set of alarms A_s):

(i) Find the earliest occurring alarm from set of alarms A_s in the data. Store its t_s as t_{start}.

(ii) Find the earliest occurring a_n alarm with $t_s > t_{start}$. Store its t_s as t_{end}

(iii) Find *all* alarms (i.e. not just including A_s alarms) which have start times that satisfy:

$$t_{start} \geq t_s < t_{end}$$

(iv) Select the next earliest occurring A_s alarm with $t_s > t_{end}$. Store its t_s as t_{start}.

(v) Repeat steps (ii)-(iv) until no more A_s alarms in the data.

In this way, any time the turbine was stopped, there was an associated batch with the set of alarms that appeared during this stoppage. A sample of a batch can be seen in table 4. Each alarm is listed according to its start time, the stop category of the alarm (note that some alarms are not A_s alarms, and so have no stop category), and the associated description. The alarm codes are left out for the sake of brevity.

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

Table 4. Example of a batch of alarm sequences

t_s	Cat.	Description
24/12/2015 03:02:01	*fc*	Rotor conv. voltage fault
	fc	Invalid conv. response
24/12/2015 03:02:40	*pt*	Blade angle asymmetry
	pt	Pitch Motor Protection
24/12/2015 03:03:02	*pt*	Blade braking time high
24/12/2015 13:06:22	*ma*	Safety Chain Pulled
	az	Yaw motor over temp.
	pt	Pitch Comms Error
24/12/2015 13:06:57	*pt*	Pitch malfunction
24/12/2015 13:26:58	*pt*	Pitch System Test
24/12/2015 13:33:23	N/A	Idling Command
24/12/2015 13:46:16	N/A	Start-up command
24/12/2015 13:48:34	N/A	Spinning Up
24/12/2015 13:49:57	N/A	System OK

azimuth bearing/rotor alarms. These “triggered” alarms are not related to the initial fault, but are instead reactions to this initial fault. A detailed analysis of these types of alarm sequences can be found in [19]. The safety chain is then pulled the next day as the technician arrives 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 online. The t_e for each alarm is not included, as all the alarms are cleared once the turbine comes back online. The total duration for this batch was just over 10 hours.

Each batch has a number of “root” alarms, which are the first alarms that appear when the turbine goes down. By looking at the stop category of these alarms, the entire batch can be attributed to being because of a fault in a particular subsystem, maintenance action, grid fault, etc. For example, in table 4, the stoppage can be attributed to the frequency converter as this is the stop category of the first two alarms in the batch. In some cases, where there are alarms with different stop categories in each set of root alarms, certain priority is given to some alarms. E.g. if a *ma* alarm appears along side some *pt* alarms, the stop is attributed to being because of a manual stop rather than a pitch fault. By associating batches like this, we can identify the cause and duration of all stoppages on the turbine.

In a previous study (submitted for publication), the authors performed a detailed analysis of these alarm batches and demonstrated how they can be used to reduce the strain of alarm analysis on turbine operators. Also detailed is the methodology for building a picture of which alarms are associated with which stop category when documentation is not available from the OEM.

The total duration of stoppages for each category of stop in the data is shown in figure 1. As can be seen, the stoppages related to pitch faults caused the most amount of downtime, with grid-related stoppages coming a close second. Because pitch stoppages are the most common type of fault, we focus on trying to predict these faults in this study.

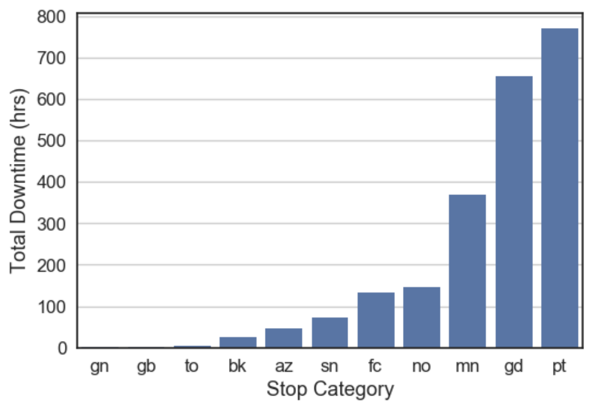

Figure 1. Total duration of stoppages for each category of stop

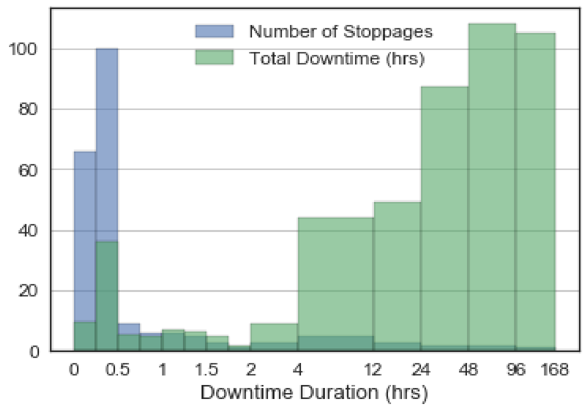

Figure 2. Distribution of pitch stoppage durations (note log scale on x axis)

A histogram showing the distribution of pitch fault durations can be seen in figure 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.

3.2. Suitability for Labelling Data
It was found that 69% of batches with a duration greater than 4 hours and stop category relating to a sub-system fault were represented in the maintenance logs as unplanned repairs, and in all these cases were correctly attributed to the correct sub-system. This was further confirmed by the availability data, where there are counters for planned and unplanned maintenance (though the root cause/sub-system is not given in the availability data, and there is no distinction between grid-related faults or turbine sub-system faults, etc.). All routine maintenance activities that appeared in the maintenance logs were also represented as such in the batch data.

Of the remaining batches that represented stoppages with > 4 hour durations and with a stop category relating to sub-system faults which were *not* represented in the maintenance logs, 23% were immediately preceded by a shorter-duration batch relating to either a grid fault or routine maintenance. The remaining 8% of batches were related to stoppages for a number of turbines on a single date which was due to a system software upgrade.

As well as stoppages which appeared in the maintenance logs, many shorter term (i.e. less severe) stoppages which did not appear in the logs were correctly captured. These were confirmed by cross-referencing with the availability data and SCADA power and wind speed values to confirm that the turbine was down during these periods.

Following these findings, a modification was made so that batches which followed each other in quick succession (i.e. with less than 2 hours difference between the end of one batch and the start of the next) were treated as one continuous stoppage, with the stop category being attributed to the first batch. Additionally, batches which occurred on dates that any software updates were carried out were removed from the data.

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.

4. Fault Prediction Methodology
4.1. Overview
Next, we attempt to predict the previously identified stoppages represented by the batches of alarms. The classification stage follows a typical machine learning strategy, outlined in figure 3. First, the batches are created using the alarms data. Next, these batches are overlaid on the 10-minute SCADA data, and times when the turbine was down due to the various types of faults (e.g. pitch, frequency converter, etc.), routine maintenance, grid issues, weather, or due to normal operation (cable untwisting, system tests), are identified.

Since we are only looking at pitch faults, batches with other types of faults, as well as times when the turbine was down due to grid issues, maintenance and stoppages as part of normal operation, are removed from the data for clear classification. Weather-related stoppages are kept in, as the wind and other weather-related parameters should inform when the turbine is down for these reasons, so the classifier should classify these times as fault-free. Hence, we are now left with fault-free SCADA data and pitch fault SCADA data. Batches below a certain minimum duration can be excluded from the data if this improves prediction performance. This would leave only faults that required a maintenance visit to restart the turbine, which may have a better fault signature in the data due to equipment degradation leading up to the stoppage. Fault-free SCADA samples from a certain time before the fault leading up to the point immediately preceding a fault sample are labelled as "pre-fault" data. This window of time leading up to the fault is known as the "pre-fault window" and can be varied to see if prediction performance improves.

Finally, the classifier is trained and tested in a variety of scenarios; namely, varying the minimum batch duration and pre-fault window, to evaluate performance.

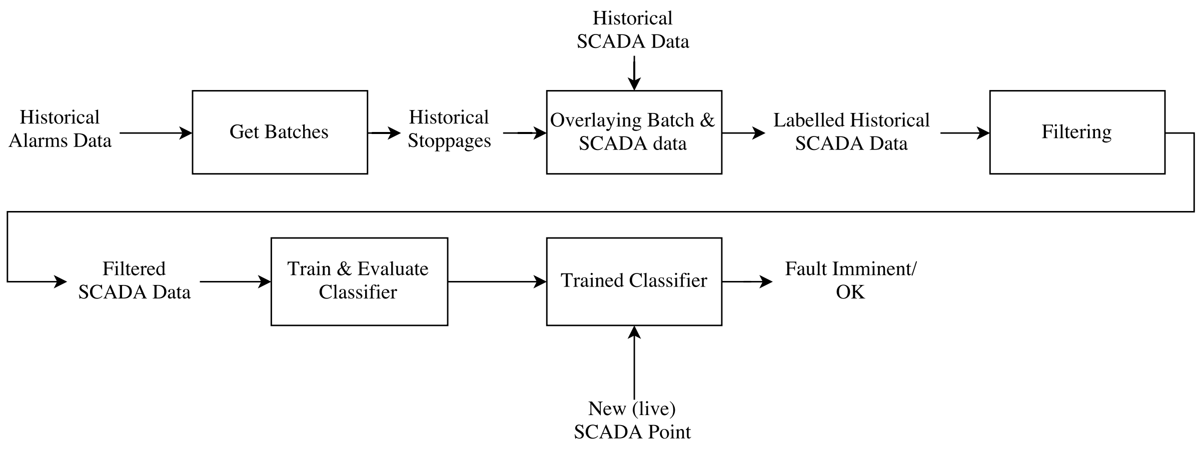

Figure 3. Overview of classification methodology

4.2. Classification Algorithm
A number of different classification algorithms were initially used on a first pass test, 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 similarly, but the decision trees were much quicker to train and test 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 [20].

4.3. Feature Selection
The set of features from the SCADA data for the classification stage was based on the features selected in [14, 21, 13], with some additions. This amounted to the following features:

- Ambient temperature
- Wind speed (average, max. & standard deviation)
- Pitch system battery box temperatures
- Pitch control box temperatures
- Average set angle, actual angle and standard deviation of actual angle for each blade
- Set and actual torque
- Nacelle orientation angle

Additionally, there were the following computed features:

- The average and standard deviation of all actual blade angles
- The absolute difference between each pair of actual blade angles (e.g. blade 1 and blade 2)
- The average and standard deviation of pitch control box and battery box temperatures
- The absolute difference between each pair of control and battery box temperatures
- The absolute difference between actual and set blade angle for each blade and torque

Note that all angles were converted to radians and broken into their sine and cosine components. The absolute difference were computed from these.

4.4. Training and Testing

Models were trained for various different minimum batch durations (0, 1, 2, 4, 8, 24 and 48 hours) and fault prediction windows (2, 4, 8, 24 and 48 hours). In the training stage, the fault class was randomly undersampled such that the number of fault samples was equal to the number of fault-free samples. A cross validation strategy was used to evaluate performance whereby all but one turbine were used for training and one turbine was held out for evaluation, meaning the test set score was an average of performance across 11 different models. The testing stage of the cross validation was done in two ways; in the first, the full set of data for the evaluation turbine was used, i.e. including the class imbalance seen in real life. The second again randomly undersampled on the test set in order to benchmark against other work done in the literature. It should be noted, however, that caution should be used when directly comparing scores against previous work done as using the same methodology on different datasets can give very different scores.

5. Results

The classification performance was evaluated using both precision and recall. The formulae for calculating these can be seen below:

$$Recall = tp/(tp + fn)$$

$$Precision = tp/(tp + fp)$$

tp is the number of true positives, i.e., correctly predicted fault samples, fp is false positives, fn is false negatives, i.e., fault samples incorrectly labelled as no-fault, and tn is true negatives.

The results for all classification strategies are shown in figure 4. As can be seen, the general trends showed that the shorter the minimum duration of the pitch-fault related batches, the higher the score generally was. This implies that, because more batches were included when there was a lower minimum duration cut-off, the improved scores here are due to more fault-class data being available for training. This is consistent with findings in [13] which showed that scores were significantly improved with more data.

It should also be noted that the mean precision score was much higher when the test sets were undersampled, which is to be expected. Recall scores were not affected as much, but were still somewhat sensitive to this. On the fully sampled test set, precision scores across all minimum batch durations dramatically increased with increasing pre-fault window. This implies that precision increased with more available data. Recall seemed to be less sensitive to the window size, though in most cases in the fully sampled test set, scores were higher for shorter windows.

The scores in general were not high enough to warrant this system being deployed in the field, with a maximum precision of 0.23 and maximum recall of 0.24 on the fully sampled test set. Ideally, a high precision score would be needed to justify installing such a system - the cost to install would be relatively low as there is little to no additional hardware needed, so a low recall score and high precision score means that although not all stoppages are being predicted, a low false positive rate means that there is still a net benefit to installing the system. With a maximum of 0.23, precision was not high enough to warrant field deployment.

The precision scores were similar to scores obtained in [13] on the fully sampled set. However, scores for both precision and recall on the undersampled set were much lower than those found in [12] and [13]. This could be down to a number of factors which warrant further investigation. It is possible that the different datasets used across different studies in literature have faults that are inherently easier or harder to predict due to different turbine models or operating conditions, or due to the specific faults that occurred in the periods being analysed having stronger leading fault signatures. It should be noted, however, that when the data was shuffled before training and testing in the undersampled case, scores dramatically increased, with precision and recall in

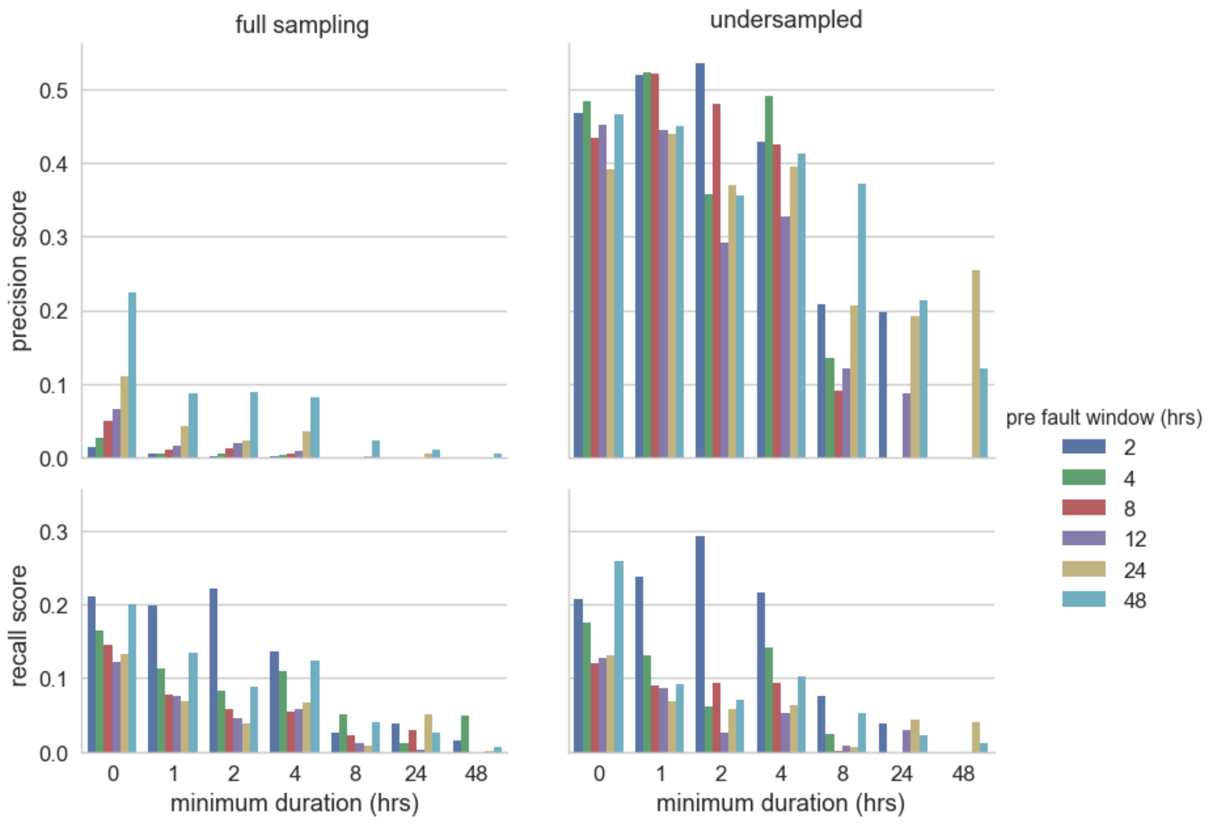

Figure 4. Overview of precision and recall scores for different sampling strategies

the high 80's and 90's. This increase is most likely due to heavy time-series effects being present in the data which artificially inflate scores. It is not clear if this step was taken in other works, which may explain the difference in scores despite similar classification approaches being used. Further investigation is needed to rule this out.

6. Conclusion

This work describes a new method to automatically identify and label every stoppage that occurred across all turbines on a wind farm using alarm system and SCADA data. This avoids the tedious step of needing to manually label stoppages in the SCADA data for fault prediction using maintenance logs. The stoppages were attributed to faults in particular subsystems, scheduled periodic maintenance events, grid-related issues, severe weather events or due to things like cable untwisting, system tests, etc. From cross referencing with maintenance logs, it was found that 92% of unplanned maintenance activities and 100% of planned maintenance activities in the logs were correctly caught. Of the unplanned maintenance activates caught, all were correctly attributed to the sub-system where the fault occurred. As well as this, many shorter-term stoppages were captured, confirmed by the turbine's availability data. Further investigations will be carried out to enable the stoppage-labelling process to be more robust to edge cases.

Fault prediction was then performed on this data using classification techniques, with various scenarios considered according to duration of the stoppages and fault prediction windows. Precision scores generally were on par with those demonstrated in literature, however recall scores were lacking. Performance dramatically increased when the data was shuffled, but these scores were artificially high due to strong time series effects.

The next steps to be taken to improve predictive performance involve creating time-lagged features to store "memory" of previous turbine states in the data. As well as this, a broader range of classification techniques will be employed to see if scores can be improved. More data will be used to see if scores can approach those seen in other works. It would be useful to benchmark all the classification techniques in literature on a unified dataset. This would also serve to rule out whether time-series effects arising from shuffling data in the cross-validation or testing stage may be inflating scores. Using the automated fault labelling process described in this research should significantly speed up the data preprocessing steps when performing classification going forward.

Acknowledgement
This research was funded by Science Foundation Ireland (SFI) Centre MaREI - Centre for Marine and Renewable Energy (12/RC/2302).

References

[1] S. Krohn, P.E. Morthorst, and S. Awerbuch, *The Economics of Wind Energy.* European Wind Energy Association, 2009.
[2] P. Tchakoua *et al.*, "Wind Turbine Condition Monitoring: State-of-the-Art Review, New Trends, and Future Challenges," *Energies*, vol. 7, no. 4, pp. 2595–2630, apr 2014.
[3] V. A. Peters, A. B. Ogilvie, and C. R. Bond, "Continuous Reliability Enhancement for Wind (CREW) Database : Wind Plant Reliability Benchmark," Tech. Rep. September, 2012.
[4] Peter Tavner, *Offshore Wind Turbines: Reliability, availability and maintenance.* Institution of Engineering and Technology, jan 2012.
[5] Eirgrid, "Industry Guide to the I - SEM," Tech. Rep., 2017.
[6] W. Yang *et al.*, "Wind turbine condition monitoring: Technical and commercial challenges," *Wind Energy*, vol. 17, no. 5, pp. 673–693, 2014.
[7] W. Yang, R. Court, and J. Jiang, "Wind turbine condition monitoring by the approach of SCADA data analysis," *Renewable Energy*, vol. 53, pp. 365–376, may 2013.
[8] J. Tautz-Weinert and S. J. Watson, "Using SCADA data for wind turbine condition monitoring a review," *IET Renewable Power Generation*, vol. 11, no. 4, pp. 382–394, mar 2017.
[9] E. Lapira *et al.*, "Wind turbine performance assessment using multi-regime modeling approach," *Renewable Energy*, vol. 45, pp. 86–95, 2012.
[10] S. Butler *et al.*, "A feasibility study into prognostics for the main bearing of a wind turbine," *2012 IEEE International Conference on Control Applications*, pp. 1092–1097, oct 2012.
[11] A. Zaher *et al.*, "Online wind turbine fault detection through automated SCADA data analysis," *Wind Energy*, vol. 12, no. 6, pp. 574–593, sep 2009.
[12] A. Kusiak and W. Li, "The prediction and diagnosis of wind turbine faults," *Renewable Energy*, vol. 36, no. 1, pp. 16–23, 2011.
[13] 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, p. 11, 2013.
[14] A. Kusiak and A. Verma, "A data-driven approach for monitoring blade pitch faults in wind turbines," *IEEE Transactions on Sustainable Energy*, vol. 2, no. 1, pp. 87–96, 2011.
[15] K. Leahy *et al.*, "Diagnosing wind turbine faults using machine learning techniques applied to operational data," *2016 IEEE International Conference on Prognostics and Health Management (ICPHM)*, pp. 1–8, 2016.
[16] R. L. Hu *et al.*, "Using Domain Knowledge Features for Wind Turbine Diagnostics," in *2016 15th IEEE International Conference on Machine Learning and Applications (ICMLA).* IEEE, dec 2016, pp. 300–307.
[17] Y. Qiu *et al.*, "Wind turbine SCADA alarm analysis for improving reliability," *Wind Energy*, vol. 15, no. 8, pp. 951–966, nov 2012.
[18] C. Kaidis, B. Uzunoglu, and F. Amoiralis, "Wind turbine reliability estimation for different assemblies and failure severity categories," *IET Renewable Power Generation*, vol. 9, no. 8, pp. 892–899, 2015.
[19] E. Gonzalez, M. Reder, and J. J. Melero, "SCADA alarms processing for wind turbine component failure detection," *Journal of Physics: Conference Series*, vol. 753, no. 7, p. 072019, sep 2016.
[20] L. Breiman, "Random Forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, nov 2001.

[21] B. Chen *et al.*, “Wind turbine SCADA alarm pattern recognition,” *IET Conference on Renewable Power Generation (RPG 2011)*, pp. 363–368, 2011.