

IMPERIAL COLLEGE LONDON

DEPARTMENT OF EARTH SCIENCE AND ENGINEERING

Wind Turbine Fault Prediction and Maintenance Scheduling: The Use of Machine Learning Techniques

Author:
Hanchao Chen

Supervisor:
Matthew Piggott
Co-supervisor:
James Percival

Abstract

Because of the significant Operation and Maintenance (O&M) cost of wind farm, there have been growing interest in developing new maintenance strategy in the offshore wind industry. Condition-based maintenance (CBM) strategy, which base on the information provided by Condition Monitoring Systems (CMS), predicts component failures and help farm owner schedule maintenance in advance. However, the high cost of additional sensors installation, personnel and software leads to limited reliability and cost-effectiveness. On the other hand, Supervisory Control and Data Acquisition (SCADA) system, which is basic component of wind turbine, is installed in almost every large wind farm to record turbines' main parameters. For this reason, the possibility of analysing the information provided by SCADA system to detect abnormalities has gained more attention in recent years. The aim of this thesis is to investigate how machine learning algorithms can be applied to offshore wind turbine operational SCADA data to predict failures and estimate remaining useful life. We build three different machine learning models, named regression model, binary classification model and advanced multinomial classification model. And we also developed a novel data pre-processing procedure which is proved to be efficient. By applying models to three turbine failure data, we conclude that our advanced multinomial classification model is able to detect different types of component failures more than one month before they were detected by purpose-designed monitoring system. Our work develops a standard workflow which can be easily applied on all kinds of SCADA data. With low demand on additional sensors, our model gives very high prediction accuracy. It has potential to be applied in real O&M work and help reduce its cost.

Contents

1	Introduction	1
1.1	Background	1
1.2	Project Purpose	1
1.3	Literature Review	2
2	Code metadata	2
3	Introduction to Neural Networks	3
3.1	Artificial Neural Networks	3
3.2	Recurrent Neural Networks	4
4	Overview of SCADA data	5
4.1	Data Acquisition	5
4.2	Data Visualization	6
4.3	Underlying Information of Data	6
4.4	General Steps of Data Pre-processing	6
4.5	Model development	8
5	Regression Model	9
5.1	Data Pre-processing	9
5.2	Model Development	10
5.3	Result and Discussion	11
6	Binary Classification Model	11
6.1	Data Pre-processing	11
6.2	Model Development	12
6.3	Result and Discussion	12
7	Advanced Multinomial Classification Model	13
7.1	Data Pre-processing	13
7.2	Model development	14
7.3	Result and Discussion	15
8	Conclusion and Future Work	16
8.1	Conclusion	16
8.2	Future work	17
A	Appendix: LSTM Networks	18
B	Appendix: Optimization algorithm	19

1 Introduction

1.1 Background

The wind is a clean, free, and readily available renewable energy source. And wind turbines allow us to harness the power of the wind and turn it into electricity. With the growing demand for environment protection, wind power generation is playing an increasingly important role in recent decades. For example, wind energy had a total net installed capacity of 169 GW in Europe in 2017. And this number became 205GW in 2019. Wind energy has become one of the fastest-growing energy sources in the world.

Wind farm is a cluster of many individual wind turbines and can be classified into onshore wind farm and off shore wind farm. Offshore wind energy, which is much more recent and emerged mainly due to less impact on landscape and to take advantage of the steadier and stronger offshore winds, has been price-competitive with conventional power sources in Europe since 2017. And the European Commission expects that offshore wind energy will be of increasing importance in the future [1].

With technology development, there is a tendency that companies increase the rated capacity of wind turbines as well as the distance between offshore wind farm and shore, to maximise the extracted power. However, this significantly increases the complexity of the project. And the demand for technological improvement not only with respect to the wind turbines themselves but also with their transportation, installation, grid integration and Operation and Maintenance (O&M) is ever increasing [19]. Compared to installation whose cost is fixed at the beginning and can be hardly controlled by wind farm owners, O&M is more flexible and has potential to be optimized during the whole life cycle. O&M accounts for 25-30% of the lifetime cost of offshore wind farms including component failures and replacement [19]. It is largely because of the remoteness of the farms, the non-optimal maintenance strategy and the harsh ocean environment they are exposed to. In order to stay competitive with respect to other rapid growing technologies, companies need to maximize the economic effectiveness of these offshore wind farms. Therefore, reducing the O&M costs becomes the main goal in the offshore wind industry. Now more and more advanced maintenance strategies are being researched and implemented by wind industry to minimize O&M costs.

Basically, there are two types of maintenance strategies: reactive maintenance and preventive maintenance. Reactive maintenance is repairs that carried out after a component failure has occurred. While preventive maintenance is regularly carried out or based on certain predefined criteria to lessen the likelihood of a failure. Although no initial cost and maintenance plan required for reactive maintenance, there are several disadvantages: more cost of unexpected downtime, shorter asset life expectancy and difficulty of time management. Since reactive maintenance is a short-sighted approach, relying only on reactive maintenance is not economic effective for the long term projects. For the wind but increasingly also other industries like, mineral processing, oil & gas - reactive maintenance is no longer a solution. This is the rationale why the wind industry pays more attention to the search on preventive maintenance. Preventive maintenance can reduce the cost and repair time but has also indirect advantages - extended lifetime and the optimization of energy production. In this context, data mining approaches which utilize the available turbine Supervisory Control and Data Acquisition (SCADA) data is being studied [11]. SCADA systems are currently being installed in every large wind farms, and therefore utilising their data involves no additional costs. SCADA system is able to monitor operating status of components and produce alarms if there are abnormal values. However these alarms are too frequent which lead to so many false positives. Hence the SCADA system is often ignored by wind farm owners.

1.2 Project Purpose

The aim of this project is to investigate how machine learning algorithms can be applied to offshore wind SCADA data and make decisions about the health status of turbines. The following are key objectives of this project:

- Determine the health state of different components of wind turbines.

- Define thresholds of machine health.
- Build a standard workflow for health prediction and maintenance decision.

1.3 Literature Review

There are four main approaches named trend analysis, clustering, normal behavior modelling and physical models, as was reported in the review by [20] with the use of SCADA data for wind turbine condition monitoring. Normal behavior modelling (NBM), which does not require detailed knowledge on the relationship between fault development and the operating parameters, gets the most attention. A NBM predicts the value of some SCADA variable (e.g. component temperature), as a function of other SCADA variables. The error is defined as the difference between the predicted values and the real values. Then a maximum value for this error is set, and error above this threshold are considered as the occurrence of turbine failure. To estimate NBM, many works have been done with the use of machine learning techniques. Here we can divide these works in two groups based on the ML approach they used: unsupervised and supervised learning [18]. As for supervised learning, labels are required and model is trained to learn the mapping function from the input variables to the output labels. Unsupervised learning methods do not require pre-existing labels. One important thing need to be clarified is that supervised learning methods are always required to create normal behavior models, and the labels are the values of target variable of NBM [10]. However, strategies where no labels indicating the turbine's operating states (e.g. normal or abnormal) are used are regarded as unsupervised methods [13].

- Unsupervised approaches

Zaher et al [22] developed a simple ANN with one hidden layer containing 3 neurons to model gearbox temperature evolution based on three months of SCADA data. The model are able to produce alarms of gearbox faults up to six months before they were detected by traditional monitoring systems. Schlechtingen et al [17] developed a linear regression model and an ANN to detect bearing damages and stator temperature anomalies. A five standard deviations limit for residuals is set. The result shows ANN generally performs better than linear regression model. Tautz et al compared four approaches named ANN, linear regression, SVM and RNN. The result shows that RNN produces similar result as ANN and ANN type approaches generally outperform other approaches. Wang et al [21] used auto encoder to predict blade breakage. The SCADA variables are first encoded by an ANN with two hidden layers. Then the compressed data are reconstructed and compared with initial data. Large value of reconstruction error indicates an abnormal state. They reported that all cases of blade breakage were detected at least six hours before they were detected by traditional monitoring systems.

- Supervised approaches

In terms of fault supervised approaches, fault detection model is usually a binary classifier distinguishing between normal state and abnormal state. The challenge is that it's hard to get ground truth labels of SCADA data. Generally, there are fewer researches on a supervised approach. Kusiak et al directly used status codes from SCADA system as labels to carry out system health monitoring. With the use of ANN they achieved the detection of faults one hour before occurrence with an accuracy between 63% and 78% [16].

2 Code metadata

Platform

The main developing language this project is Python3. In addition, the libraries below are included. For the developing platform, the software is developed under Mac OS and Windows10. The machine learning models are run and tested on author's Macbook pro. The platform for machine learning is Jupyter Notebook, and the GPU used is one NVIDIA RTX 2070 8GB.

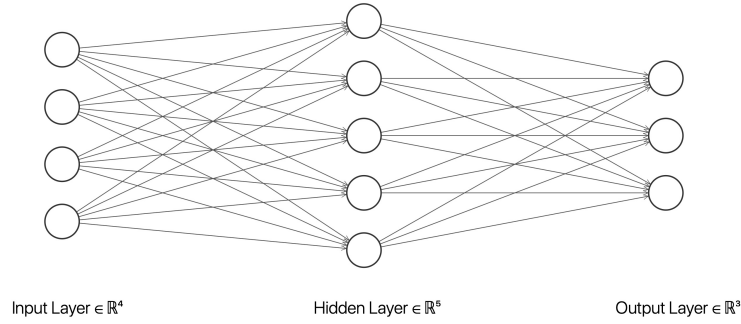


Figure 1: Simple fully connected neural network

Libraries

The following are utilised in this work:

1. Pytorch (1.5)[5]
2. TensorFlow[7]
3. Pandas[4]
4. Numpy[3]
5. Matplotlib[2]
6. scikit-learn[6]

Link to codes

<https://github.com/acse-2019/irp-acse-hc1119>

3 Introduction to Neural Networks

3.1 Artificial Neural Networks

Artificial Neural Networks (ANN) [12] are bionics network structures inspired by the performance of the human brain. Its basic idea is to build complex network structure by connecting a large number of simple neurons, which allows signals to be transmitted from one neuron to another. Signals are amplified or attenuated by activating different neurons and adding weights.

3.1.1 Fundamental of ANN

Figure 1 shows a traditional ANN structure:

- Input layer: contains several neurons and receives multi-dimension input signals.
- Output layer: contains several neurons and produces multi-dimension output signals.
- Hidden layer: contains several layers and each layer contains several neurons.

When signals are fed to certain neuron, they are multiplied by weights of the connections and added together. Then the weighted sum is passed through an activation function to produce the output. Some commonly used activation functions are listed in table 1.

Number of layers, number of neurons, connection between neurons and activation functions are main factors that determine a network's performance. We can build different ANN by using different choices of these factors. An ANN model's performance can be quite different when the model is applied on different kinds of input signals. There is no universal ANN model that performs well on all kinds of signals.

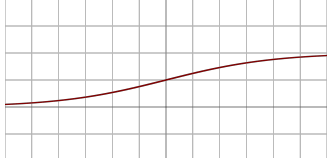
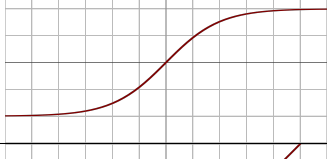
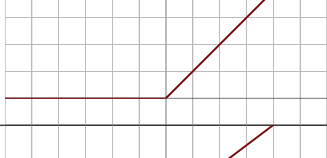
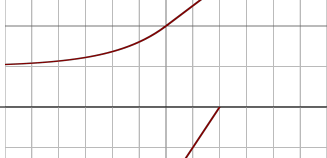
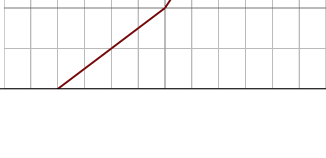
Activation Function	Equation	Plot
Sigmoid Function	$f(x) = \frac{1}{1+e^{-x}}$	
Hyperbolic Tangent Function	$f(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$	
Rectified Linear Unit Function	$f(x) = \begin{cases} x=0 & \text{for } x \leq 0 \\ y=x & \text{for } x > 0 \end{cases}$	
Leaky Linear Unit Function	$f(x) = \begin{cases} x=0.01x & \text{for } x < 0 \\ y=x & \text{for } x \geq 0 \end{cases}$	
Exponential Linear Unit Function	$f(x) = \begin{cases} x=\alpha x & \text{for } x < 0 \\ y=x & \text{for } x \geq 0 \end{cases}$	

Table 1: Common activation functions

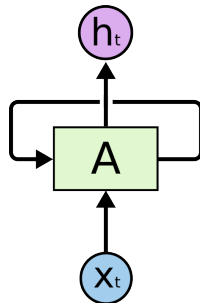


Figure 2: Recurrent Neural Networks

3.1.2 Shortcomings of traditional ANN

In traditional fully connected neural network, signals are transmitted from input layer to the output layer in sequence. And there's no interaction between neurons in the same layer. Here comes a problem that the output signal only depends on the input signal, and has nothing to do with the order of the input signal. Moreover, the neuron itself does not have the ability to store information, and the entire network does not have the "memory" ability. When the input signal is time-related, for example a time series, the performance of traditional neural network structure will be very poor.

3.2 Recurrent Neural Networks

As was stated in previous section, traditional neural networks don't perform well on time-related data. Recurrent neural networks (RNN) address this issue. They are special networks with loops in them, allowing information to persist.

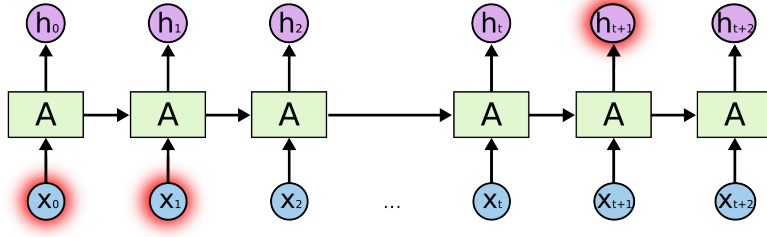


Figure 3: The problem of long-term dependencies

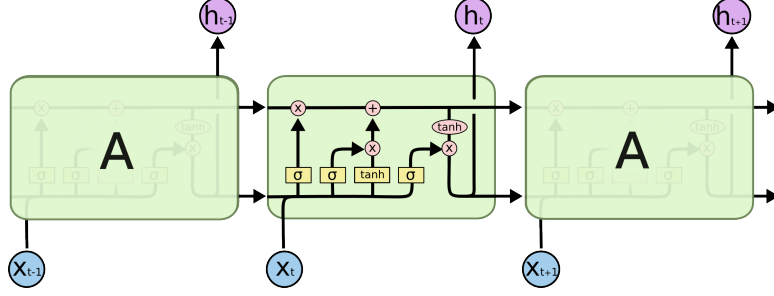


Figure 4: Structure of LSTM

3.2.1 Structure of RNN

Figure 2 shows the compact structure of RNN, where x_t is input signal, h_t is output signal and A is a chunk of neural networks which allows information to be passed from one step of the network to the next. To make it clear, a RNN can be regarded as multiple copies of the same network, each passing a message to a successor.

3.2.2 Shortcomings of RNN

One of the advantages of RNNs is that they might be able to connect previous information to the present task. However, when the gap between the relevant information and the point where it is needed is very large, for example a long time series data, RNNs become unable to learn their relationship.

3.2.3 Long Short Term Memory networks

Long Short Term Memory networks (LSTM) are a special type of RNN, which are capable of learning long-term dependencies. They were invented in 1997, and were developed and popularized by many people in following work. LSTM networks have same chain like structure as RNN, but the repeating module has a different structure. Repeating module in RNN only has a single neural network layer while there are four hidden layers which interact with each other in very special way in the repeating module of LSTM. Detailed introduction are shown in Appendix A.

3.2.4 Conclusion

In this section, the basic concepts required to develop a neural network models and the comparison between traditional ANN and LSTM has been introduced. The LSTM model is more capable of dealing with time series data which means it is a suitable method for this project.

4 Overview of SCADA data

4.1 Data Acquisition

All available SCADA parameters compiled in a typical wind turbine are presented in table 2.

Categories	Variable name
Environment	Ambient temperature
	Wind speed
Status	Current
	Energy Export
	Gear oil temperature
	Generator RPM
	Generator temperature
	Grid Frequency
	High speed bearing temperature
	Nacele position
	Power
	Reactive power
	Rotor speed
	Voltage

Table 2: Available SCADA parameters

4.2 Data Visualization

Before further transformation of data set, visualization techniques are used to extract the main characteristics and trends. Pandas library provides simple method to visualize the correlation between parameters. A correlation matrix of the selected variables was shown in figure 5 and the plots in the diagonal are density plots of each of the variables.

If one variable presents a linear correlation with the other variable, one of them can be simply dropped since it doesn't provide any additional information in the model.

4.3 Underlying Information of Data

There are two different ways of understanding the SCADA data. One is to regard the data as a continuous time series, the other one is to regard them a several discrete data point. Both way make sense, but different understandings lead to different models.

For the first one, a regression model can be built to predict the values of chosen parameters. Error is defined as the difference between the predicted values and the real values. Error above certain value is considered as the occurrence of turbine failure.

For the second one, a classification model can be built based on every single data point. For example, a binary classification model can simply use two labels indicating the turbine's state (e.g. "normal" and "abnormal"). Multinomial classification model is also possible.

4.4 General Steps of Data Pre-processing

Data pre-processing is an important step in the data mining process. There are always missing values, duplicate values, out-of-range values and impossible data combinations, etc. Analyzing data that has not been pre-processed can produce misleading results. Data pre-processing transforms raw data into meaningful information. Thus, it is considered as the most important phase of a machine learning project.

The following pre-processing steps are general steps which were applied to the data sets we used for all three models.

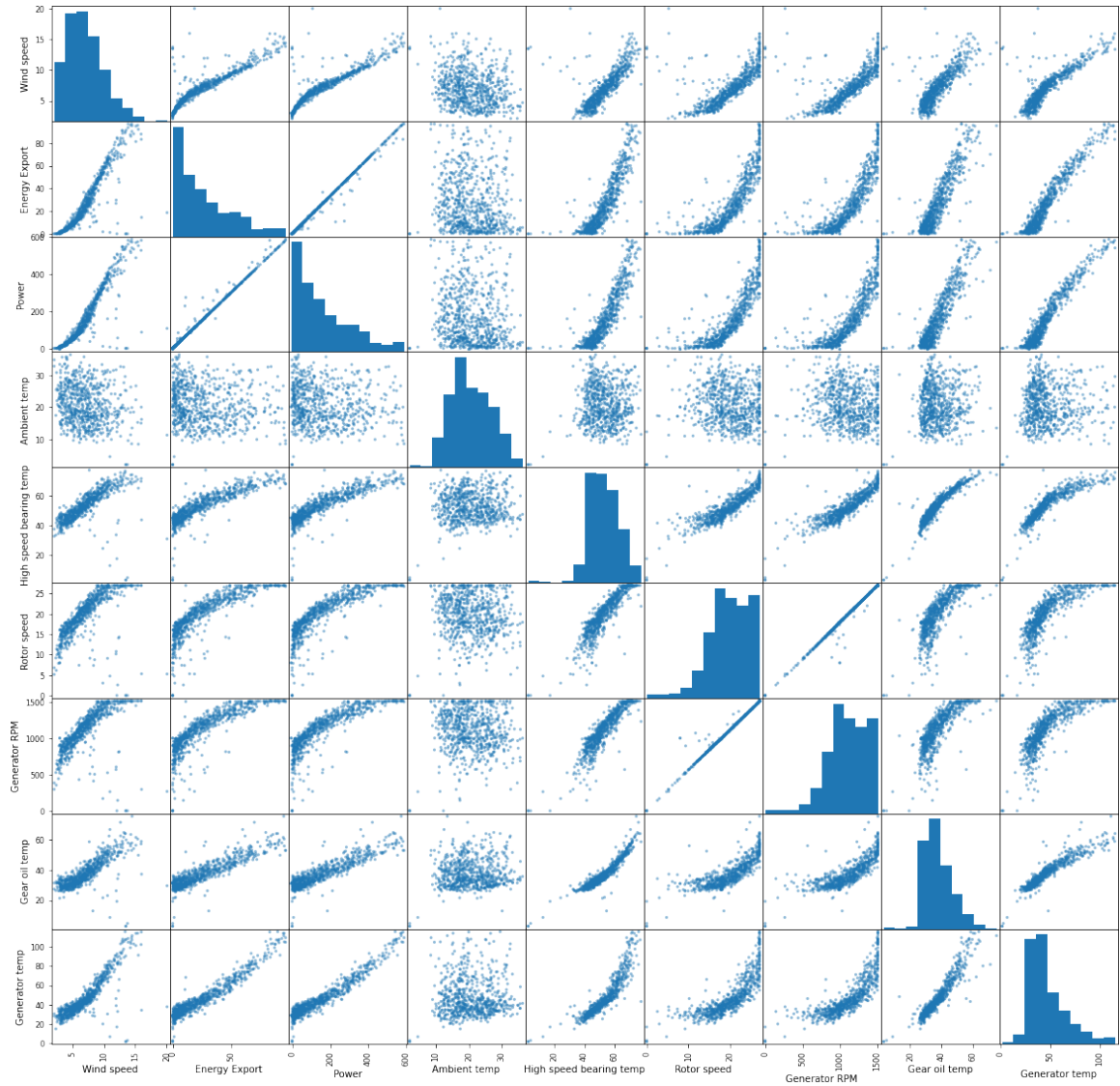


Figure 5: Correlation matrix of the studied variables

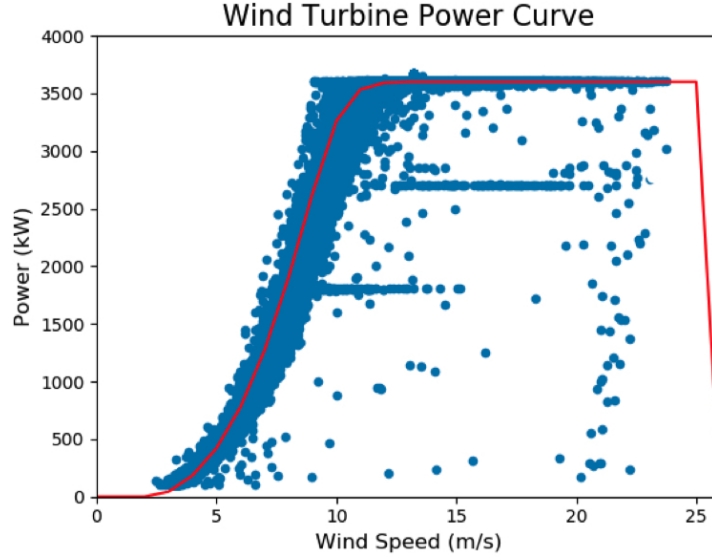


Figure 6: "Power-Wind speed" curve of turbine [19]. Each blue dots represents a 10-minutes SCADA data point, red line is the theoretical power curve

1. **Missing values removal:** Delete all rows include missing values by using the command "dropna" in pandas library.
2. **Low output power values removal:** Delete rows where output power values are lower than 10kW. Since these low power output values represent transient states or forced stopped states which do not provide useful information.
3. **Curtailement and outliers removal:** For a wind turbine, there's a theoretical relationship between output power and wind speed.

$$P = \frac{1}{2} \rho \pi R^2 C_p u^3$$

where ρ is the air density, R is the radius of wind turbine rotor, C_p is power coefficient and u is wind speed. A turbine's power-wind speed curve can be plotted using this equation. However, the cut-in speed and rated power also influence the curve in real case. The cut-in speed is the minimum wind speed required for the wind turbine to start Operating and the rated power output is the maximum power output the wind turbine can reach [19]. A typical power-wind speed plot is shown in the following figure, where the red line is the theoretical power curve. Most of the SCADA data points follow the curve line fairly accurately, while there are some outliers far away from the curve. These outliers do not correspond with the normal behaviour states. All we need to do is remove rows corresponding to these outlies.

4. **Data normalization:** Scale all the variables to the same range to eliminate the scale effect. The following formula is used:

$$x' = \frac{x - \text{mean}(x)}{\text{std}(x)}$$

where $\text{mean}(x)$ is the average value of variable x , $\text{std}(x)$ is standard deviation of variable x .

4.5 Model development

Figure 7 shows general steps to build neural networks.

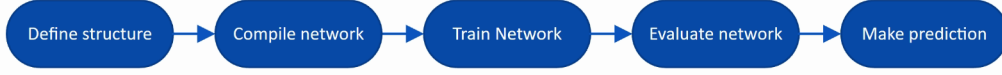


Figure 7: General steps to build neural networks

4.5.1 Define structure

The first step is to decide the type of the neural network. Then the detailed structure parameters should be defined. Activation function should also be defined if necessary.

4.5.2 Compile network

The second step is to arrange the sequence of layers. The network will be trained and evaluated through a series of matrix transforms. Furthermore, the optimization algorithm and loss function need to be defined in this step. Detailed introduction on optimization algorithm is shown in Appendix B. For this study, a Adaptive Moment Estimation (Adam) algorithm was used for the optimization.

4.5.3 Train Network

Training set has already been defined in data pre-processing section. Training set is used to update the weights of neurons to create a good mapping of inputs to outputs, and an optimization algorithm is also involved here. In this step, epoch number and batch size are defined.

The number of epoch determines how many times the optimization algorithm will work through the entire training set during the training process. Too few epochs will result in an under-fitted model while too many epochs will lead to an over-fitted model. Both of them produce inaccurate result when presented with new data.

The batch size is the number of training examples utilized in one iteration. Grid search algorithm can be used here to find an optimal value of batch size. The maximum value of batch size is also limited by the available memory of computer.

4.5.4 Evaluate network

In this step, the network was evaluated on the evaluation set, to test its performance on unseen data. Usually evaluation set is part of training set. After evaluation process, training set is combined with evaluation set to form a complete training set. Network are trained again on this larger training set.

5 Regression Model

It's natural to start with developing a simple regression model of unsupervised approach. One simple regression model was an ANN with one fully connected hidden layer containing 10 neurons [19]. The output is continuous numeric value. Then the health status is decided based on the difference between predicted values and real values. The majority of works on regression model use traditional ANN and RNN, but few of them use LSTM which have better performance on time series data. In this section, we tried to substitute the simple fully connected neural network with the LSTM network.

5.1 Data Pre-processing

In addition to the general data pre-processing steps, some extra data pre-processing steps also need to be done to make data suitable for our LSTM-based regression model.

1. Variables selection

Input and output variables selections are mainly based on the type of failure. As for a gearbox failure, we can choose certain gearbox component as output variable (target variables)[16], and

Input Variables	Output Variables
Power	gear oil temperature
Wind Speed	
Ambient Temperature	
gear oil temperature	
High speed bearing temperature	

Table 3: Variables selection

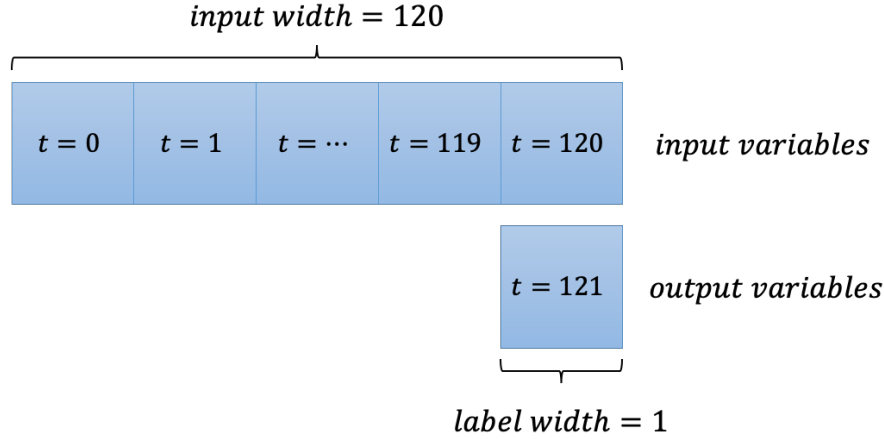


Figure 8: The window of input data

related environmental variables as input variables. In this case, our goal is to build model to predict gearbox failure. And we used the following choice shown in table 3.

Note that the autoregressive model was used here, where historical values of the target variable were used to build the model and predict the target output[20].

2. Data windowing

The regression models make a set of predictions based on a window of consecutive samples from the data. The main feature of the input windows need to be decided is the width (number of time steps) of the input and label windows. For example, to make a prediction of output variable 1 step (10 minutes) into the future, given 120 steps (20 hours) of history of input variables, a window should be defined like this:

3. Data set division

For training and validating the model, the data set was divided into three parts, training set, validation set and test set. Since the objective of regression model is to predict target variables' value of normal operating state, one year data whose end date is 3 months ahead of failure occurrence was chosen as the training period set. Then the training period data set was randomly split into a training set (80% of data set) and a testing set (20% of data set). From 3 months ahead of failure occurrence to the point of failure occurrence, data of abnormal state was chosen as the test set.

5.2 Model Development

The keras library provide simple implementation of LSTM model. By setting the "return_sequence" as false and a proper batch size, a LSTM model is completely built. And the procedure of training and evaluating the model has been demonstrated in previous section.

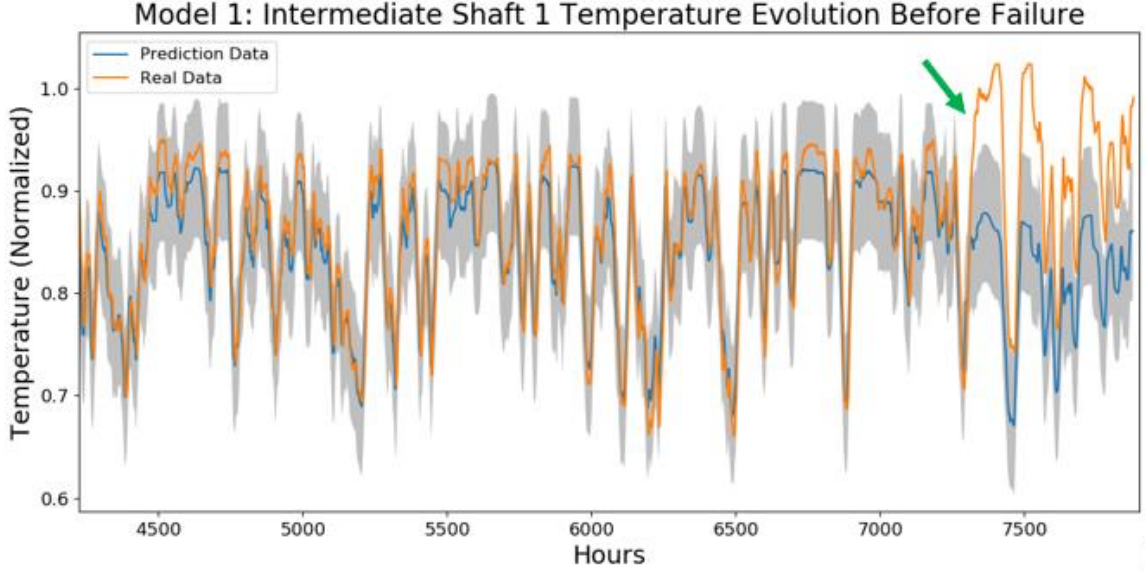


Figure 9: Intermediate shaft 1 temperature evolution [19]. Blue line: predicted temperature evolution, orange line: real temperature evolution. Grey region represents the prediction interval.

5.3 Result and Discussion

It can be seen from figure 9 that the predicted values exactly follow the fluctuations of the real values. This means that the LSTM-based regression model is able to extract the underlying time-related information of SCADA data. There is a considerably deviation between predicted values and real values in the time interval indicated by two red arrows. This means that the regression model was able to predict the incipient failure approximately 3000 hours before it was detected by the other monitoring system.

However, a target variable which has direct relation with the failure is required for this method (e.g. gear oil temperature in this case). The incipient component failures should have obvious influence on the value of this variable. Following a regression model[19], intermediate shaft 1 temperature was used as output variable to predict the gearbox failure. Figure 9 shows that there is a significant increase of this value before the occurrence of failure. In this project, only two variables related to gearbox failure were provided. It is a common problem that a failure can not be represented by one variables. To address this issue, a different strategy need to be developed to make use of all available variables.

6 Binary Classification Model

With the consideration of limited SCADA variables, supervised approaches which do not require a direct failure related variable was developed.

6.1 Data Pre-processing

1. **Variables selection** This binary classification model does not require a direct failure related variable, so that all environmental variables (e.g. wind speed, ambient temperature) and state variables (e.g. gear oil temperature, generator temperature) can be used as inputs. In this case, in order to predict a failure, three environmental variables (wind speed, ambient temperature) and three state variables (power, gear oil temperature and high speed bearing temperature) were selected.
2. **Labels definition** Labels representing two classes are required for the binary classification model. The time period of 6 months which is 5 months ahead the failure detection is defined as normal state. And the time period of 1 month just before the failure detection is defined

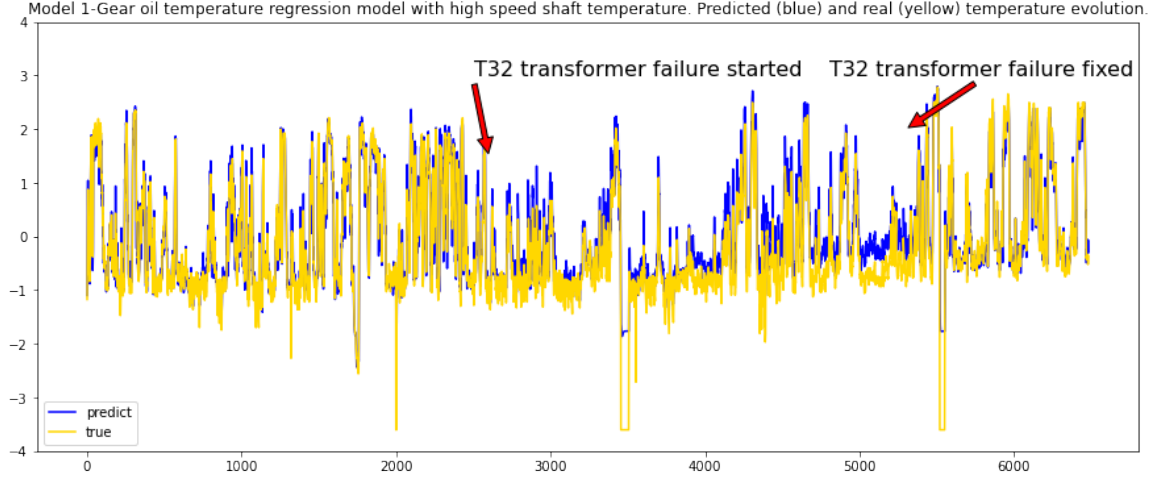


Figure 10: Gear oil temperature evolution during turbine LLY-21 failure development. Blue line: predicted temperature evolution, yellow line: real temperature evolution. Red arrows represent important events of failures

Categories	Failure and maintenance	Start data
Failure	T32 transformer failure	2019-07-24
Preventive maintenance	TX replacement (T17)	2018-06-28
	TX replacement (T25)	2018-09-24
	Gear oil change	2018-10-10
	TX replacement (T26)	2019-01-09
	T16 transformer exchange	2019-09-24
	Gear oil cold or superheated	2019-12-16

Table 4: Status record of wind turbine LLY-21 (2018.06.01-2019.12.30)

as failure state. Data points under normal state were labeled with "0", and data points under failure state were labeled with "1".

3. **Data set division** All data points were randomly split into a training set (80% of data set) and a testing set (20% of data set).

6.2 Model Development

A simple ANN with two fully connected hidden layers was built as a classifier. There are 100 neurons in the first hidden layer, 50 neurons in the second hidden layer. The values of these two hyper-parameters are carefully chose to achieve good performance on detecting highly non-linear relationships in data with low computational cost. The output layer contains 2 neurons because the label contains two classes. ReLU function was chosen as the activation function. And logsoftmax function was used in the output layer, as one hot-encoding was used to define each class. With these numerical outputs, the prediction accuracy is defined as the number of correctly predicted states divided by the number of all outputs for a testing period.

6.3 Result and Discussion

According to the status record, "T32 transformer failure" was detected on 24-Sep-2019. In addition to component failure, several maintenance was also recorded. They were supposed to be preventive measure to avoid failures. Details are shown in table 4.

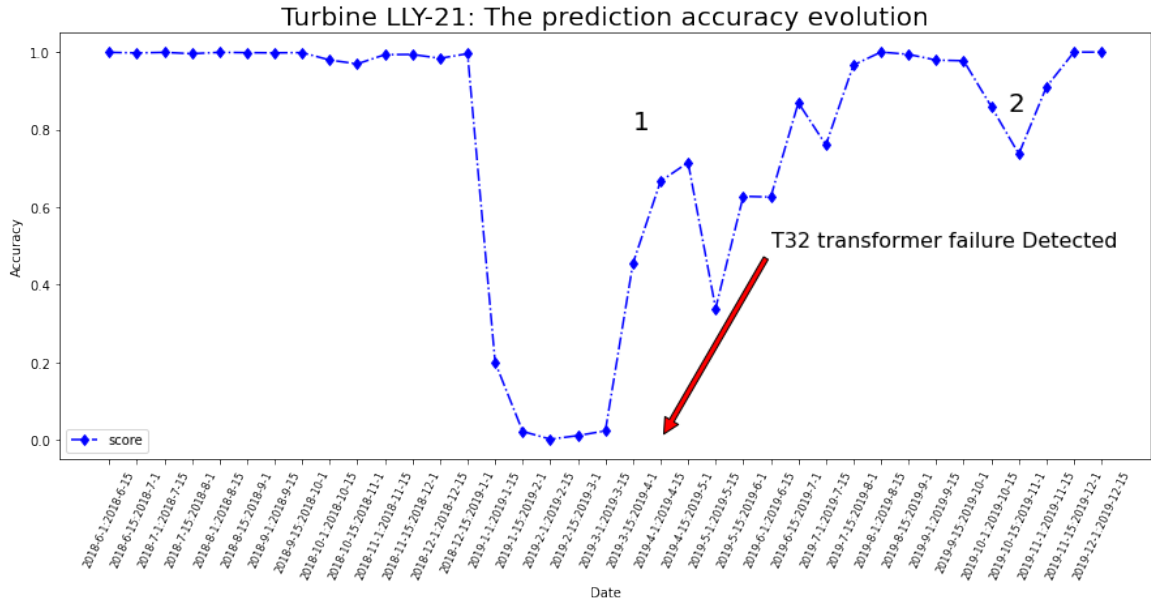


Figure 11: Turbine LLY21-Binary classification model. Prediction accuracy (blue line) evolution from 2018-6 to 2019-12.

It can be seen from the figure 11 that the huge trough 1 corresponds to "T32 transformer failure" and trough 2 corresponds to "Gear oil problems". However, there were two recorded maintenance carried out in the flat part of the blue line which corresponds to training data set. The model obviously ignore these events. This can be explained by the model over-fitting. Since there were not enough related variables (e.g. bearing temperature corresponds to gearbox failure), the model was easily over-fitted during training process.

In addition, the binary classification model simple used all selected variables without considering the underlying relationship environmental variables and status variables. More features of the data need to be extracted before the training process.

7 Advanced Multinomial Classification Model

An advanced multinomial classification model was developed to address the issues mentioned above. In order to avoid over-fitting, the model was designed to be general failure predictor which was not purpose-designed for specific failure.

7.1 Data Pre-processing

1. **Variables selection** Two environmental variables and all four failure-related variables were selected even though they were related to different types of failures.
 - Ambient Temperature
 - Power
 - Gear oil temperature
 - Generator RPM
 - Generator temperature
 - High speed bearing temperature
2. **Division of operating state** Cui et al [9] introduced a method of removing outliers in SCADA data. The data were separated into clusters based on the wind speed and production power.

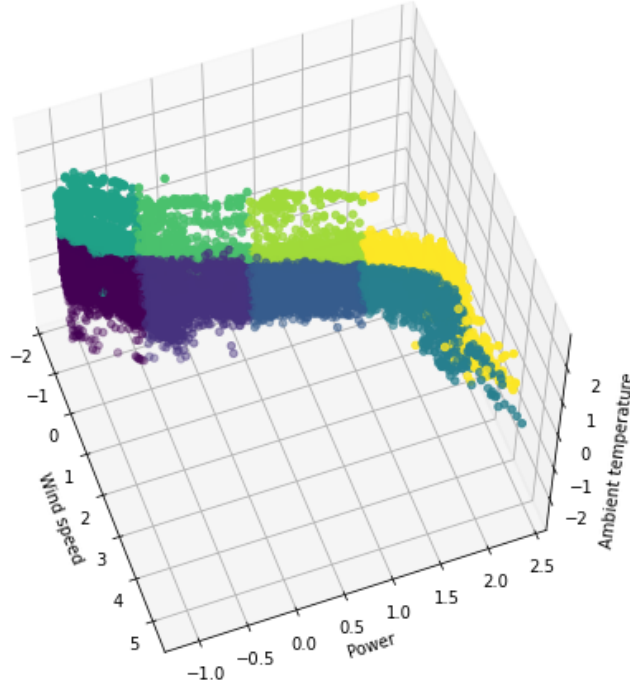


Figure 12: Division of operating state. Each cluster with certain color represents one operating state. The values of three variables are normalized values.

Points which located outside three standard deviations from the mean value in each cluster by Mahalanobis distance were assumed to be outliers and were deleted by the filter. With the consideration of this, variables above were divided into two groups, "active variables" and "passive variables". Active variables include ambient temperature, power and wind speed while passive variables include gear oil temperature, generator RPM, generator temperature and high speed bearing temperature. Then data were separated into clusters based on three active variables. Each cluster represented an operating state. Once the values of active variables and the operating states were given, the range of passive variables' values would be fixed. In this case, operating state of the turbine is divided into 8 classes according to three active variables. Each class of operating state was labeled with a number from 0 to 7.

3. **Data set division** All data points were randomly split into a training set (80% of data set) and a testing set (20% of data set).

7.2 Model development

As mentioned above, operating states have direct relation with passive variables. Given values of passive variables, their operating states can also be predicted. A multinomial classification model was developed here to do this job.

7.2.1 Define structure

In this case, a simple neural network with two fully connected hidden layers was designed. There are 100 neurons in the first hidden layer, 50 neurons in the second hidden layer. The number of neurons in the input layer equals to the number of input variables which is 4. Since there are 8 different operating states, the output layer contains 8 neurons. The number of neurons was chosen through trial and two hidden layers were used to reduce the training complexity of the models. ReLU function was chosen as the activation function and logsoftmax function was used in the output layer to produce one-hot-encoding result.

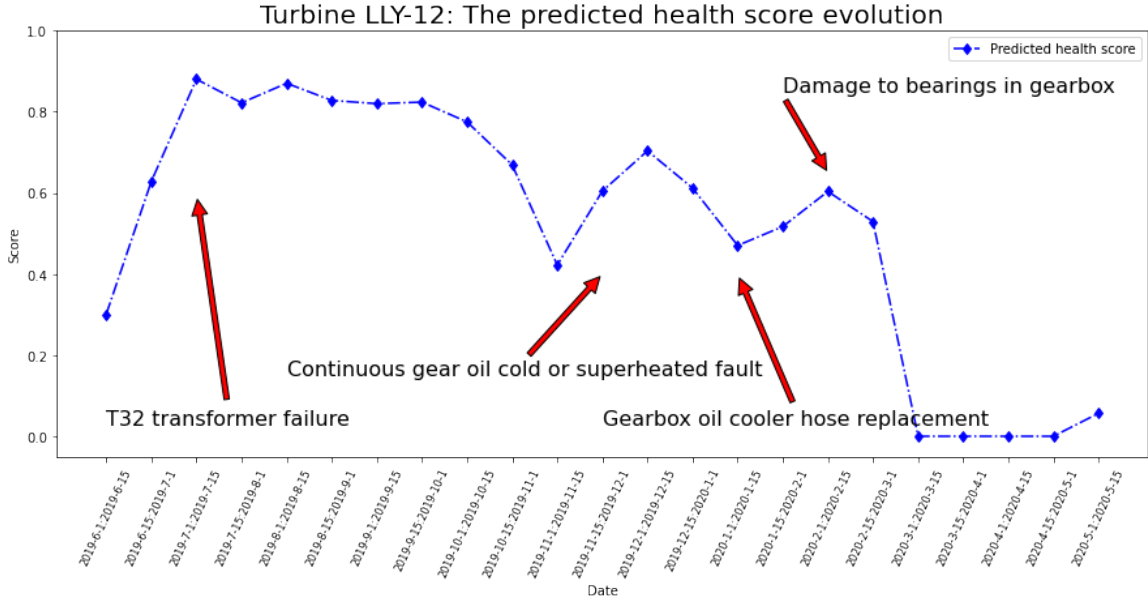


Figure 14: Turbine LLY12-Advanced multinomial classification model. Predicted health score (blue line) evolution from 2019-1 to 2020-5.

the failure. Serious failure (e.g. T32 transformer failure) leads to very low health score and "Replace" alarm was triggered. The other alarms can be explained as early stage failures and the growth of these failures were stopped by preventive maintenance.

All abnormal states of turbine LLT-21 were detected by this model according to health score evolution graph (figure 11) and status record (table 4). In terms of the major serious transformer failure of LLY-21, there is a "Replace" alarm triggered more than 2 months before the outage. Early stage failures are also detected by this model. Turning now to turbine LLY-12, it can be seen from figure 12 that the model ignored some failures. This is because this model is not expected to detect all types of failures with the consideration of limited input variables. Even so, this model is able to detect major failures listed in table 5.

Since this advanced multinomial classification model is not purpose-designed for specific failure, the health score is influenced by different types of failures which makes it hard for users to find out which failure the alarm corresponds to. However, given enough SCADA variables, this model can be easily changed to be a purpose-designed model for specific failure.

8 Conclusion and Future Work

This section presents the conclusions of the study and a discussion for further research.

8.1 Conclusion

The aim of this project is to investigate how machine learning algorithms can be applied to offshore wind SCADA data and make decisions about the health status of turbines. And all three key objectives are achieved in the end:

- Three different machine learning based models for predicting wind turbines' health status are developed.
- A new data pre-processing method is created.
- A standard workflow for health prediction and maintenance decision is built.

In the beginning, data pre-processing techniques were implemented for each SCADA variables. Based

on this analysis, three machine learning models were built to predict the health condition of the turbines. The sequence of the three models represents the deepening of the understanding of SCADA data. By comparing the effectiveness of these three models, there is one additional finding that exploring the underlying relationship between variables before feed them into the model is extremely important. The new data pre-processing method is created to achieve this goal. And that's why the last advanced multinomial classification model has the best performance.

Moreover, all three models are able to process one turbine's data and make prediction in 5 minutes, and only requires very few computational resources. This property shows the potential for real-time monitoring.

8.2 Future work

As was mentioned in section 5, regression model using LSTM has good performance in predicting evolution of certain variable. This means that regression model is able to extract the underlying time-related information of SCADA data. However, the two classification models use single data point as input, therefore the time-related information is not utilised. It is deserved to explore combining LSTM with the classification model. In this way, short series of data points are used as inputs and the machine learning model's ability to extracting information will be maximized.

Acknowledgements

I would like to thank Professor Matthew Piggott and Dr James Percival for their insightful guidance. Without their help, this thesis could not have reached its present form. I would also like to thank my friend Qizhi, Cai whose brilliant ideas have proved immensely constructive. Finally, my thanks would go for my parents for their unlimited support for me all the time.

References

- [1] After a decade of dithering, the US east coast went all in on offshore wind power this week. <https://en.keyenergy.it/key-energy/info/news/key-energy-news/after-a-decade-of-dithering-the-us-east-coast-went-all-in-on-offshore-wind-power-this-week.n8604504.html>, 2018.
- [2] Matplotlib-tutorials. <https://matplotlib.org/tutorials/index.html>, 2020.
- [3] Numpy-Community. <https://www.numpy.org>, 2020.
- [4] Pandas-tutorials. <https://pandas.pydata.org/docs/>, 2020.
- [5] PyTorch-Community. <https://pytorch.org>, 2020.
- [6] Scikitlearn-guide. https://scikit-learn.org/stable/user_guide.html, 2020.
- [7] Tensorflow-Community. <https://tensorflow.google.cn/community>, 2020.
- [8] L. Bottou. Stochastic gradient descent tricks. In *Neural networks: Tricks of the trade*, pages 421–436. Springer, 2012.
- [9] Y. Cui, P. Bangalore, and L. B. Tjernberg. An anomaly detection approach based on machine learning and scada data for condition monitoring of wind turbines. In *2018 IEEE International Conference on Probabilistic Methods Applied to Power Systems (PMAPS)*, pages 1–6. IEEE, 2018.
- [10] J. Eriksson. Machine learning for predictive maintenance on wind turbines: Using scada data and the apache hadoop ecosystem, 2020.
- [11] J. L. Godwin and P. Matthews. Classification and detection of wind turbine pitch faults through scada data analysis. *IJPHM Special Issue on Wind Turbine PHM*, page 90, 2013.
- [12] S. Haykin and N. Network. A comprehensive foundation. *Neural networks*, 2(2004):41, 2004.
- [13] G. Helbing and M. Ritter. Deep learning for fault detection in wind turbines. *Renewable and Sustainable Energy Reviews*, 98:189–198, 2018.

- [14] N. Kanwar et al. *Deep Reinforcement Learning-based Portfolio Management*. PhD thesis, 2019.
- [15] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.
- [16] A. Kusiak and W. Li. The prediction and diagnosis of wind turbine faults. *Renewable energy*, 36(1):16–23, 2011.
- [17] M. Schlechtingen and I. F. Santos. Comparative analysis of neural network and regression based condition monitoring approaches for wind turbine fault detection. *Mechanical systems and signal processing*, 25(5):1849–1875, 2011.
- [18] S. Shalev-Shwartz and S. Ben-David. *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014.
- [19] W. Sofia. Condition-Based Maintenance of Offshore Wind Farms: The Use of SCADA Data in Normal Behaviour Modelling. Master’s thesis, Imperial College London, 2018.
- [20] J. Tautz-Weinert and S. J. Watson. Using scada data for wind turbine condition monitoring—a review. *IET Renewable Power Generation*, 11(4):382–394, 2016.
- [21] L. Wang, Z. Zhang, J. Xu, and R. Liu. Wind turbine blade breakage monitoring with deep autoencoders. *IEEE Transactions on Smart Grid*, 9(4):2824–2833, 2016.
- [22] A. Zaher, S. McArthur, D. Infield, and Y. Patel. Online wind turbine fault detection through automated scada data analysis. *Wind Energy: An International Journal for Progress and Applications in Wind Power Conversion Technology*, 12(6):574–593, 2009.

A Appendix: LSTM Networks

The key element of LSTM is the cell state C_t which is the horizontal line running through the top of the diagram. Present information is recorded in cell state and sent to next step of LSTM. It’s like a belt runs through the entire network and keeps the information all the way. And there are structures in LSTM called gates that have ability to remove or add information to the cell state. LSTM has three kinds of gates, named forget gate, input gate and output gate.

Just as its name implies, forget gate decides what information will be thrown away from the cell state. And the operation is done by a simple sigmoid layer. Forget gate takes h_{t-1} and x_t as inputs, and outputs a number between 0 and 1 for each number in the cell state C_{t-1} . For example, 1 means "completely keep this" while 0 means "completely forget this".

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f)$$

Input gate is a combination of a sigmoid layer and a tanh layer which decides what new information will be stored in the cell state. The sigmoid layer generate a value i_t between 0 and 1 decides which values to be update. The tanh layer creates a vector of new candidate values \tilde{C}_t . Then \tilde{C}_t is multiplied by i_t to update the cell state.

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)$$

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C)$$

Finally, the output gate decides the output value h_t . The output is based on cell state which has been modified by forget gate and input gate. First, a sigmoid layer is added to select input value. Then, the cell state will go through a tanh layer and multiplied by the output of the sigmoid layer.

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o)$$

$$h_t = o_t * \tanh(C_t)$$

B Appendix: Optimization algorithm

Gradient descent is a first-order iterative algorithm to minimize the loss function. At the beginning, the weights (θ) of the neurons are set randomly. Their values will be updated every epoch based on the gradient of the loss function $J(\theta) = \frac{1}{m} \sum_{i=1}^m L(\theta, (x^i, y^i))$:

$$\theta = \theta - \eta \nabla \theta J(\theta; x^i; y^i)$$

where η is the learning rate. The loss function are function of difference between estimated and true values for an instance of data. The gradient descent function calculates the gradient for each of the samples and averages the values[8]. It is one of the simplest algorithm but very time consuming and inefficient for large data sets. SGD calculates the gradient for only one randomly chosen example in each case.

Adam algorithm is a advanced version of SGD algorithm using the combination of momentum method and RMSProp method. The algorithm is shown below:

$$\begin{aligned} J(\theta) &= \frac{1}{m} \sum_{i=1}^m L(\theta, (x^i, y^i)) \\ \mathbf{s} &= \rho_1 \mathbf{s} + (1 - \rho_1) \mathbf{J} \\ \mathbf{r} &= \rho_2 \mathbf{r} + (1 - \rho_2) \mathbf{J}^T \mathbf{J} \\ \hat{\mathbf{s}} &= \frac{\mathbf{s}}{1 - \rho_1} \\ \hat{\mathbf{r}} &= \frac{\mathbf{r}}{1 - \rho_2} \\ \Delta \mathbf{J} &= -\epsilon \frac{\hat{\mathbf{s}}}{\sqrt{\hat{\mathbf{r}} + \delta}} \end{aligned}$$

In terms of Adam algorithm[14], estimations of first and second moments are both utilised. It is useful to think of a ball running down a slope. Adam behaves like a heavy ball with friction, which prefers stopping at flat minimum in the error surface [15].

Adam algorithm has better capability to avoid stopping at local minimum with the use of momentum, and it also avoid the accumulation of second order momentum which can lead to a early stop of the training process. It is a relative universal algorithm which performs well on both dense and sparse data set.

Turning now to the second part, the loss function used here to assess the accuracy of the network is the Mean Squared Error (MSE), which can be defined as:

$$MSE = \frac{1}{N} \sum_{i=1}^N (f_i - y_i)^2$$

where N is the number of samples in data set, f the predicted target value and y the real target value. Low loss value in the training process does not necessarily indicate an accurate model. An over-fitted model also produce low loss value.