Energy and AI 4 (2021) 100065

Contents lists available at ScienceDirect

Energy and AI

journal homepage: www.elsevier.com/locate/egyai

Autoencoder-based anomaly root cause analysis for wind turbines

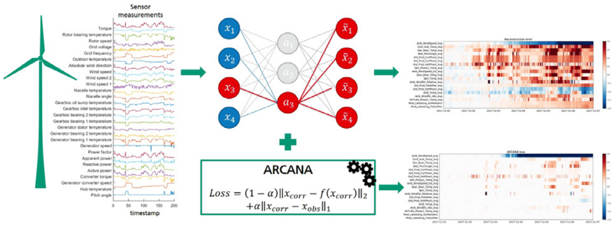
Cyriana M.A. Roelofs[a], Marc-Alexander Lutz[a,*], Stefan Faulstich[a], Stephan Vogt[a,b]

[a] Fraunhofer IEE, Königstor 59, Kassel 34119, Germany
[b] Intelligent Embedded Systems, Universität Kassel, Wilhelmshöher Allee 67, Kassel 34121, Germany

HIGHLIGHTS

- A method for autoencoder-based anomaly root cause analysis (ARCANA) is introduced.
- The method provides human interpretable explanations for anomalies.
- The explanations help to identify the underlying root causes.
- The applicability of ARCANA is demonstrated on wind turbine sensor data.
- Validation of the explanations is shown with failure data.

GRAPHICAL ABSTRACT



ARTICLE INFO

Article history:
Received 17 December 2020
Received in revised form 25 February 2021
Accepted 1 March 2021
Available online 5 March 2021

Keywords:
Anomaly detection
Autoencoder
Root cause analysis
Predictive maintenance
Wind turbine
Explainable artificial intelligence

ABSTRACT

A popular method to detect anomalous behaviour or specific failures in wind turbine sensor data uses a specific type of neural network called an autoencoder. These models have proven to be very successful in detecting such deviations, yet cannot show the underlying cause or failure directly. Such information is necessary for the implementation of these models in the planning of maintenance actions. In this paper we introduce a novel method: ARCANA. We use ARCANA to identify the possible root causes of anomalies detected by an autoencoder. It describes the process of reconstruction as an optimisation problem that aims to remove anomalous properties from an anomaly considerably. This reconstruction must be similar to the anomaly and thus identify only a few, but highly explanatory anomalous features, in the sense of Ockham's razor. The proposed method is applied on an open data set of wind turbine sensor data, where an artificial error was added onto the wind speed sensor measurements to acquire a controlled test environment. The results are compared with the reconstruction errors of the autoencoder output. The ARCANA method points out the wind speed sensor correctly with a significantly higher feature importance than the other features, whereas using the non-optimised reconstruction error does not. Even though the deviation in one specific input feature is very large, the reconstruction error of many other features is large as well, complicating the interpretation of the detected anomaly. Additionally, we apply ARCANA to a set of offshore wind turbine data. Two case studies are discussed, demonstrating the technical relevance of ARCANA.

1. Introduction

Operation and maintenance (O&M) costs account up to 30% of the total cost of energy for offshore wind farms. As wind turbine (WT)s become more and more complex and are generally built in remote locations with harsh weather conditions, failures and resulting downtimes are common [1]. These downtimes have to be reduced to increase revenue. Therefore, the ability to detect failures early and identify the root cause becomes increasingly important. Identification of the root cause is not only valuable information for early failure detection, but also in retrospect, as it can be used to eliminate such errors in the future.

In the last couple of years much research has been conducted on data-driven health monitoring for WTs using machine learning (ML) models to address this problem [2]. Many of these data-driven approaches focus on training a normal behaviour model (NBM) to detect anomalous

[*] Corresponding author.
E-mail addresses: cyriana.roelofs@iee.fraunhofer.de (C.M.A. Roelofs), alexander.lutz@iee.fraunhofer.de (M.-A. Lutz), stefan.faulstich@iee.fraunhofer.de (S. Faulstich), stephan.vogt@uni-kassel.de (S. Vogt).

https://doi.org/10.1016/j.egyai.2021.100065

behaviour or specific failures in WT Supervisory Control and Data Acquisition (SCADA) data [3].

A popular anomaly detection method for this type of use case is to use autoencoder (AE)s, an unsupervised type of artificial neural network (ANN). Nonetheless, AE models do not show the underlying cause for the detected anomalies directly. This is however required for eliminating the cause of errors, decision making and planning of maintenance measures. An analysis method to determine what caused the anomaly is therefore needed. In this paper we introduce a novel method to address this problem and move towards human understandable ML model output. Several ML methods for such a purpose exist. This paper focuses on methods which are based on AEs. The novelties and highlights of the paper are as follows:

- A novel method for autoencoder-based anomaly root cause analysis (ARCANA) is introduced.
- The method provides human interpretable explanations for anomalies.
- The explanations help to identify the underlying root causes.
- The applicability of ARCANA is demonstrated on wind turbine sensor data.
- Validation of the explanations is shown with failure data.

An overview of related work is given in Section 2. In Section 3, we introduce the AE and its implementation to detect anomalies in WT SCADA data. Next, we introduce a method for anomaly root cause analysis called ARCANA in Section 4. The results are compared to the baseline AE method for the same use case. We show that ARCANA points out the relevant contributing features for detected anomalies, whereas using only the AE reconstruction errors does not. The technical relevance of ARCANA is demonstrated in Section 5 using two case studies based on WT SCADA data of an offshore wind farm. Finally, the results are discussed in Section 6 and summarised in Section 7.

2. Related work

An overview of the related scientific work is given in this section. Herefore the topic is divided in three aspects: AE based anomaly detection on SCADA data, explaining ML models and explaining AE anomalies.

2.1. Autoencoder-based anomaly detection on SCADA data

With the increasing amount of operational data of WTs being available, ML methods are used to avoid unplanned WT downtime by predicting possible failures in advance. Several implementations of ML methods on WT SCADA data exist [2]. Some of these methods focus on ANN to detect faults [3].

AEs are a special type of ANN with one or more hidden layers which reconstruct the input data. AEs for anomaly detection are an unsupervised ML technique, the input data does not need to be labelled and little prior knowledge about input data is necessary. When trained on normal data instances, the AE fails to reconstruct abnormal input data. Therefore a higher AE reconstruction error (RE) is seen in case of anomalies [4].

A more detailed description of the functionality of AEs is given in Section 3.1. This Section provides an overview on AEs for SCADA data based anomaly detection.

AEs can be used to detect anomalous behaviour for the WT as a whole or for a specific system or component of the WT. Usually AEs are trained with all available sensor measurements of the WT. Lutz et al. describe the usage of an AE to detect anomalies for offshore WTs [5]. By using the trained model, anomalies before historical failures could be seen in advance in the central lubrication system, the generator transformer system and the rotor system. Renström et al. applied an AE on real world data and found anomalies in the yaw encoder system as well as an increased hydraulic oil temperature [6]. Wu et al. propose a multi-level-denoising AE which is tested in a case study and compared to other approaches [7]. Jiang et al. introduce an AE with temporal information to consider temporal dependencies of the sensor variables [8]. The AE is tested on simulated and SCADA data. Zhao et al. describe an AE with an adaptive threshold method [9]. Long et al. present an approach which uses AEs to monitor blade breakage [10]. Other approaches focus on the detection of blade icing [11].

As the previous studies have shown, AEs are a promising tool for anomaly detection on WT SCADA data. Nonetheless, the AE lacks the ability to point out the sensors measurements or features responsible for the anomaly. To determine what failure or deviation caused the anomaly, a root cause analysis that indicates the feature contributions is needed. If in the following ‘root cause analysis’ is mentioned, it refers to the aforementioned definition. An overview of current solutions to this problem is outlined in the next sections.

2.2. Explaining ML models

A systematic review regarding this topic is given by Vilone and Longo [12]. An overview of different ML explainability approaches regarding feature relevance is given by Belle and Papantonis [13]. Several methods have been developed in the last couple of years to interpret predictions of ML models. A comparison of different explainable ML approaches regarding their quality of explanation for anomaly detection is currently not known to the authors. Chatterjee and Dethlefs propose a combination of a recurrent neural network and an Extreme Gradient Boosting (XGBoost) decision tree classifier for explainable anomaly prediction. The hybrid model is trained and tested on data of an offshore WT and furthermore tested on data of an onshore WT via transfer learning [14].

Marti-Puig Zhang et al. present an overview of automatic input selection algorithms for WT failure prediction. Additionally, an exhaustive-search-based quasi-optimal algorithm is introduced as a reference for feature selection. Other automatic input selection algorithms are compared to the reference [15].

Zhang et al. present a combination of random forests and XGBoost. The first of which is used to rank the features by importance and the second of which is utilised to train the classifier for certain faults [16].

Other methods, which could be used for AEs, are LIME [17], DeepLIFT [18] and SHAP [19]. The LIME algorithm explains predictions of a machine learning model f by approximating it locally around a given prediction $f(x)$ with a linear explanation model [17]. The explanation model is applied on simplified interpretable inputs x' that are easy to understand for humans.

DeepLIFT is a recursive prediction explanation algorithm specifically for deep learning [18]. It attributes contribution scores to input features as the difference from some reference input to explain the difference in the output from some reference output.

The SHAP algorithm is based on shapley value estimations and is presented by Lundberg and Lee [19]. They demonstrate that SHAP values are more consistent with human intuition than LIME or DeepLIFT. To explain a specific prediction, the SHAP method assigns an importance value to each feature called the SHAP value. SHAP values are however hard to compute exactly. There are a couple of approximation methods such as the Kernel SHAP method to compute these values more efficiently.

2.3. Explaining autoencoder anomalies

The Kernel SHAP method has been applied to AE anomaly detection by Antwarg et al [20]. However, the method explains the RE of individual features and does not explain why an anomaly based on the anomaly score (i.e. MSE or Mahalanobis distance) was detected. Since SCADA data usually contains numerous features, of which many may have a large RE, this method does not provide a simple interpretation of detected anomalies. Also, the kernel SHAP method assumes feature

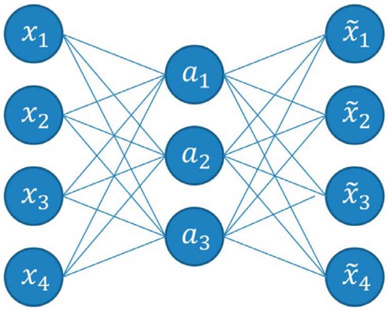


Fig. 1. A simple undercomplete AE model with only one hidden layer.

independence, which is often not the case for features used in anomaly detection.

Tagawa et al. introduce a structured denoising AE, which enables analysis of feature contributions to detected anomalies [21]. Nonetheless, it requires prior knowledge about the data structure. Specifically, the model needs to know which features depend on each other and what relationships are interesting to monitor for different types of anomalous behaviour. This is hard to implement for SCADA data which can contain hundreds of features and many different types of failures.

Another method is demonstrated by Nguyen et al., who use a variational autoencoder (VAE) to detect computer network anomalies [22]. The gradients of the VAE are used to identify the main features that caused the anomaly. If the RE changes a lot when a feature of an anomalous data point is varied by a small amount, the feature value is seen as abnormal.

Another approach within this domain is to use a sparse optimisation method, as proposed by Ikeda et al. [23]. The method finds feature contributions for AE anomalies, similarly to the ARCANA algorithm we introduce in this paper. They use an iterative proximal gradient method to optimise the loss function and the NSL-KDD dataset [24] to evaluate the method. Two novelties of this paper are to be highlighted:

First, our method is implemented using standard backpropagation, which allows for it to be implemented in the popular Keras [25] and Tensorflow [26] ANN frameworks.

Second, none of the above methods were applied on WT SCADA data and are not easily applied to our or a similar use case. In this paper, we show how to use ARCANA for any type of AE and its applicability on WT SCADA data.

Additionally, real world data often lacks labels for failures or anomalies. If there are failures known, very limited information on these failures is available. Also, it can be assumed that a wide variety of faults are possible. Even if many labelled data are available, it may be difficult to cover all eventualities with them. Therefore, only unsupervised learning techniques are considered in this paper: The AE and the proposed method ARCANA.

3. Autoencoder based anomaly detection and interpretation

In this Section the mathematical assumptions of the AE are presented, the data set is described and the anomaly interpretation using solely the AE output is shown.

3.1. Method

AE models consist of two parts: an encoder $h : \mathbb{R}^d \to \mathbb{R}^e$, which results in a compressed latent representation $a \in \mathbb{R}^e$ of the input data $x \in \mathbb{R}^d$ and a decoder $g : \mathbb{R}^e \to \mathbb{R}^d$, which decodes the latent data representation back to a reconstruction $\tilde{x} \approx x$ of the input. The RE of the AE is then given by $\tilde{x} - x$. Both the encoder and decoder should be differentiable functions and may consist of one or multiple layers. An example of a simple AE is shown in Fig. 1. The number of neurons in the input layer and the output layer are of the same size as the number of input dimensions, which is the number of SCADA data features used, e.g. the number of sensor measurements of the WT which can be used for training the model.

When trained with normal data instances (i.e. without anomalies) the AE learns to reconstruct normal data [4]. Therefore it learns to represent normal behaviour. During training, the model parameters are adjusted such that the RE is minimised, for example by minimising the mean square error (MSE) or a similar metric. In our AE set up, the MSE,

$$MSE = \frac{1}{N}(\tilde{x} - x)^T \cdot (\tilde{x} - x), \tag{1}$$

is minimised to achieve this. Here, N is the number of input features.

Many different AE architectures are seen. Considering sequential data, LSTM AEs, Gated Recurrent Unit AEs and Stacked Denoising AEs are discussed in [4]. These models make use of the temporal component of time series data. Vogt et al. have implemented an Undercomplete AE, a Denoising AE and a Contractive AE on SCADA data. Both Denoising and Contractive AEs are regularised AEs, which makes them less sensitive to small data variations. The Undercomplete AE performed best and its failure detectability is evaluated in [5]. A similar AE configuration is used in this paper.

After training, the model can be used to detect anomalies by calculating an anomaly score S_x for each input sample x. This score can be the RE, but this measure implicitly assumes that the REs of the individual features $(\tilde{x}_i - x_i)$ are independent of each other. This is typically not the case for undercomplete AEs, although most of the dependencies are captured in the latent encoding. Therefore, the Mahalanobis distance is used:

$$S_x = \sqrt{(\tilde{x} - x)^T \Sigma^{-1} (\tilde{x} - x)}, \tag{2}$$

where Σ is the covariance matrix of the training input. Since the covariance matrix of the input features is unknown, we use a robust covariance estimator [27].

Next, the anomaly score is used to define which inputs are considered abnormal by comparing it to a threshold th. If $S_x > th$, the input sample x is considered an anomaly. This threshold can for example be a certain quantile of the training data set's anomaly scores. If however there are labels available showing at which timestamps the WT was operating normally and abnormally, the threshold can be calibrated. For this, we use the same approach as in [5] and set a value for the false detection rate (FDR) to determine a proper threshold.

The AE architecture used in this paper is a fully connected neural network with three hidden layers. Different AE set ups and architectures have been tested by Vogt et al. [28]. Results on selecting the type of AE are considered to be valid in this paper as well. Architectures with up to three hidden layers have been tested. Three hidden layers showed best results. The number of neurons in the hidden layers are optimised for the data using Bayesian hyperparameter optimisation. The search space for the first and third hidden layer are between 100 and 600 neurons, the search space for the second hidden layer is between 5 and 30 neurons. Optimal results were received with layers of 400, 15 and 400 neurons. After training the mean absolute error on the training and validation data were 0.097 and 0.010 respectively.

The activation function used is the parametric rectified linear unit. The network's parameters are optimised using the Adam optimisation algorithm [29]. The FDR for anomaly threshold calibration is set equal to 0.2, which is equivalent to allowing one in five detected anomalies to be false.

Other hyperparameters that have been tuned are the learning rate and the decay rate of the Adam optimisation algorithm and the number of epochs.

3.1.1. Data preparation

Since most time series are incomplete, we need to drop features or timestamps with missing data or impute the missing data. We drop any features which are missing in more than 20% of the timestamps. Next, any remaining data gaps are filled using mean imputation. Whenever a value is missing for a specific feature, it is replaced with its mean value in the training data set. Note that a linear imputation might represent the sensor behaviour better for short data gaps. However, data gaps usually last hours up to several days. In this case the mean of the data gap is closer to the mean of all measures of the sensor and therefore more appropriate [5].

Afterwards, the data is standardised with the empirical mean and standard deviation of each feature in the training data, such that each scaled feature has a mean of zero and standard deviation of one. This generally helps in training a neural networks when inputs are on very different scales, such as measurements of different sensors.

3.1.2. Reconstruction error for anomaly interpretation

In this section, we use the elementwise RE for each input feature as indication for its contribution to a detected anomaly. These contributions can then be used to further identify the root cause. As most input features are measured on very different scales, the RE must be scaled by the standard deviation of these features to compare them among each other. The scaled elementwise RE for feature x_i is given by

$$\text{scaled RE}_i = \frac{x_i - \tilde{x}_i}{\text{std}(\boldsymbol{x_i}) + \epsilon}, \tag{3}$$

where $\tilde{x}_i$ is the reconstruction of input feature x_i and ϵ is a small number to prevent division by zero. The standard deviation is calculated based on the data within the prediction time window. Assuming that the expectation of the reconstruction error is zero and that there are no dependencies between the scaled RE_i, the sum of squared scaled REs is proportional to the exponent of a joint normal distribution of all scaled RE_i. Both assumptions are plausible, since after a successful training, the AE should not have systematic biases and should have made use of all dependencies between the input features. Therefore, the RE should be close to zero on average and there should be no correlations left between the errors of two measurements.

3.2. Data set

For this Section and Section 4, the Open Wind Farm data set from EDP [30] is used, which contains both SCADA data as well as a failure logbook for five 2 MW WTs of the same wind farm for 2016 and 2017.

The SCADA data consist of 55 different measurements, such as wind speed, temperatures inside the generator, wind direction and pitch angles. All measurements are available as ten-minute-averages.

The data is filtered based on the power generated, because we want the AE to learn normal behaviour of the WT. Therefore, only data during production is used for training, i.e. when the power > 0. The authors are aware that anomalies can be present even if the WT is in production. Nevertheless, it is expected that the WT operation is normal and according to design specifications the majority of the time when it produces power. Considering all data except the failures as normal can lead to undesired behaviour. Rare events such as technical standbys could be considered as an anomaly. However, the aim is rather to identify possible failures than to identify technical standbys.

For training the AE one year of data which is considered as normal is needed. The seasonal dependencies contained in the data can thereby be learned as well, so these will not erroneously be detected as anomalies. Smaller training periods of less than one year have not yet been tested. This method discards failures in the first year of the data set for testing, since the training data should chronologically always be before the testing data set. This greatly limits the failures that can be used for a ground truth and a possible evaluation based on the failure logbook. In addition to that, we usually do not know the exact features causing an anomaly, unless the failure is described very well. This is not the case for the failures in the failure logbook of this data set. An artificial error is therefore added on the data set.

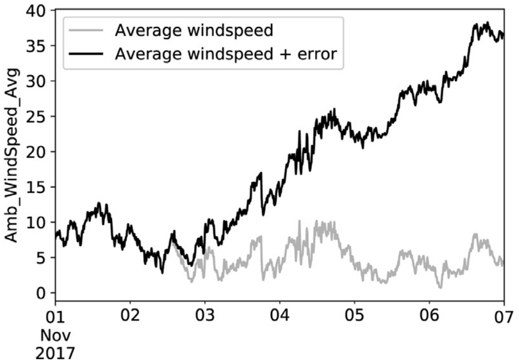


Fig. 2. The actual value of the average ambient wind speed for WT T07 and its value with added linearly increasing error.

For this experiment, this error is added to the wind speed sensor measurements during the prediction period. This enables us to determine whether the method for determining feature contributions shows the relevant feature. The artificial error added is a simple linearly increasing deviation with time from the actual measurement and is shown in Fig. 2. The artificial error was chosen so that its maximal accumulated value on the average wind speed does not exceed the WT design specifications as stated by the IEC 61400-1. The summed value of the average wind speed and the artificial error should not exceed a value of more than 50 m/s [31].

For this use case, data of WT T07 is used. For training, data between 2016-10-31 and 2017-10-31 was used, for prediction the first seven days of November 2017 which include the manipulated wind speed measurements. These dates were chosen such that there is no failure present within the prediction period according to the failure logbook. This ensures easier interpretation of the results. As mentioned above, the data is also filtered so that only instances are used for training where the power > 0.

3.3. Results

The detected anomalies for the use case are shown in the Fig. 3, as well as a heat map of the time series of the scaled REs of the 20 features with the largest REs. The anomalies detected match the period during which the wind speed sensor error was added to the prediction data. There are multiple short anomaly time windows (sequential timestamps detected as anomaly) seen due to the power production being lower than or equal to zero. As we considered this as an abnormal operation mode and removed those timestamps from the training data, any anomalies detected during this operation mode are ignored, since they are expected. They are also recognisable in the heat map, as the scaled RE gets large for some of the features during these periods as well.

The error starts on November 2nd and the REs increases from November 3rd onward. This is especially clear for the wind speed measurement. In Fig. 4 the scaled RE is shown for the anomaly for normal operational mode (OP-mode)s only. The RE for the wind speed measurement is large, however, not significantly larger than the other features shown.

Since there are many features with large average REs, it is hard to interpret the failure without looking at the individual time series of the input and reconstruction of each individual input feature with a large reconstruction error. Even then, interpretation is obfuscated by the features which are not the cause of the anomaly. It may be impossible for more complicated failures.

4. ARCANA (Anomaly root cause analysis)

As shown in the previous section, the AE is able to detect anomalies very well, but cannot show clearly which feature was the root cause of

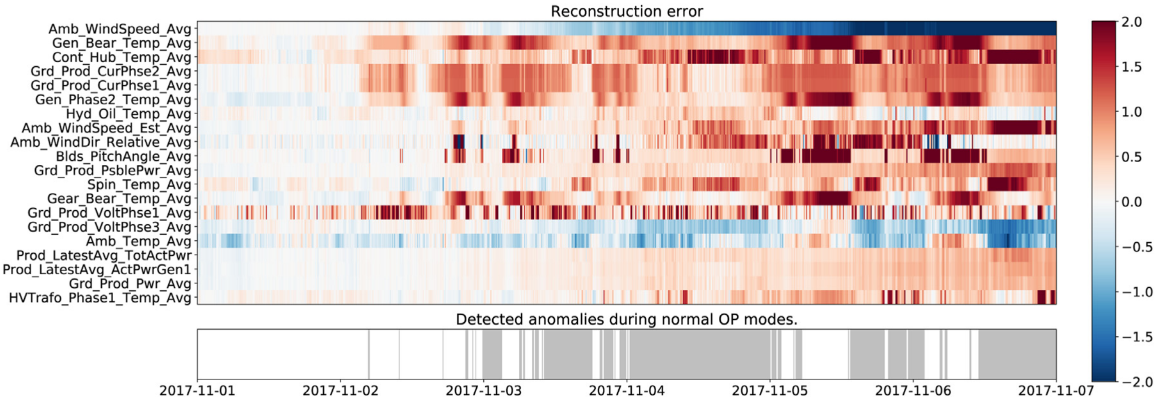


Fig. 3. Top: Heat map of the scaled reconstruction error for the prediction period with added wind speed sensor error. Only the 20 features with largest average absolute reconstruction error are shown. Bottom: Detected anomalies by the AE for the wind speed sensor error use case for WT T07. Only anomalies detected during normal operation modes are shown.

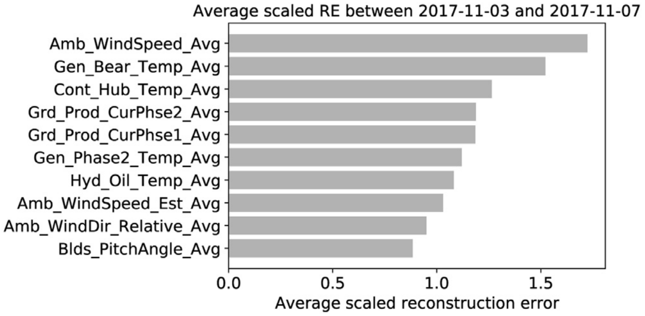


Fig. 4. Top 10 features with largest scaled reconstruction error for the anomaly time window with added wind speed sensor error.

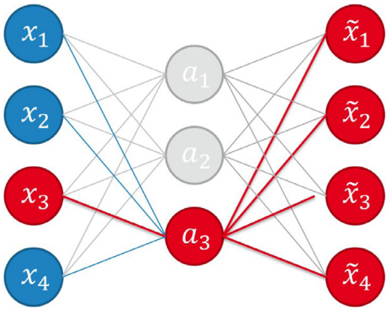


Fig. 5. An error in one input feature of an autoencoder is propagated through the network and affects all outputs of the network.

the anomaly. In this Section we introduce a novel method called ARCANA to find the relevant features. This method is applied to the same use case as in the previous section.

4.1. Method

To find the features that explain AE anomalies, we assume that anomalous behaviour of a WT usually only influences a relatively small set of SCADA measurements. Even if only one or two input features of an AE model show an error or deviation, these errors are propagated through the AE network and perturb all output values, resulting in a reconstruction error in most other features as well. This is shown in Fig. 5 for a small fully connected AE model.

To solve this problem, we propose ARCANA (Anomaly Root Cause Analysis): an optimisation algorithm to find which input features contributed to an anomaly detected by the AE model. It is based on the aforementioned assumption that WT failures are local (a specific component or sensor is broken) and therefore anomalies only influence a small set of measurements. The algorithm searches deviating input features, while trying to find only a few explanatory features that may have caused the detected anomaly. This is done by minimising the loss function

$$Loss = (1-\alpha)\frac{1}{2}\|x_{corr} - g(h(x_{corr}))\|_2 + \alpha\|x_{corr} - x\|_1, \tag{4}$$

where x is a data sample (input feature vector at some timestamp), $\alpha \in (0,1]$ is a hyperparameter, $x_{corr} = x + x_{bias}$ and x_{bias} is the ARCANA bias vector. The bias vector represents the deviation of each feature causing the anomaly and therefore their contributions to the anomaly. The bias vector may also be interpreted as the 'corrected' RE. Optimisation of the left L_2-norm finds a 'corrected' input x_{corr} which would not result in an anomaly. The L_1-norm ensures that the number of non-zero features of the bias vector is as low as possible. The hyperparameter α can be picked depending on the data and use case. For α close to 1, only the most important features are corrected. For α close to 0 the bias vector will contain as many features as needed to find a non-anomalous input, which of course is not desired for our use case.

The bias vector x_{bias} is found by minimising the loss function 4. To optimise the loss function we apply backpropagation to calculate the gradients of the loss function with respect to x_{bias}. To update x_{bias} and for the optimisation the Adam [29] algorithm is used. An initial guess for x_{bias} may be the RE, zero or a weighted version of these two. For this paper we used the RE as initial guess for x_{bias} and $\alpha = 0.8$.

Note that x_{bias} is calculated for each sample, just like the anomaly score and the RE. The bias vector is used to calculate the ARCANA feature importances or anomaly feature contributions. This is done by averaging the absolute bias vector values over a certain time window (i.e. an anomaly time window) and normalising the result, such that the sum of the importances is equal to 1.

4.2. Data set

The same data set is used as described in Section 3.2 for the RE method for anomaly interpretation. To demonstrate the ARCANA algorithm, we use the same set up for the AE model and use the same use case as in Section 3. For prediction, the ARCANA algorithm is applied to find the feature contributions.

4.3. Results

In Figs. 6 and 7 the ARCANA results for the wind speed sensor error are depicted for the top 20 features with the highest ARCANA importances. The features shown differ slightly from the features shown in Figs. 3 and 4.

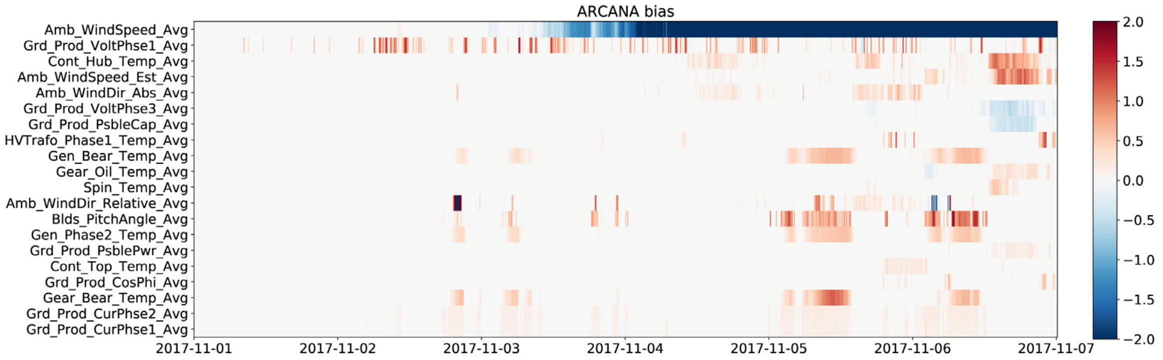


Fig. 6. Heat map time series of the ARCANA bias for the prediction period with added wind speed error. Only the 20 features with the largest ARCANA importance are shown.

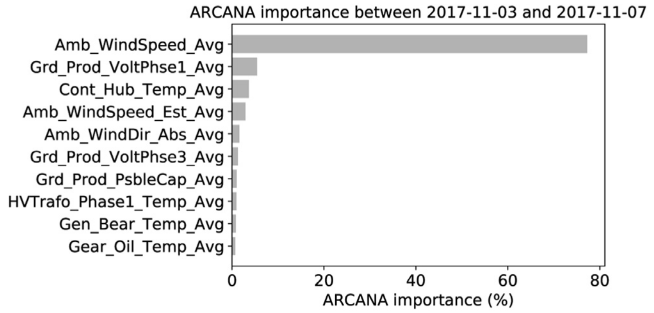


Fig. 7. Top 10 features with largest ARCANA importances for the anomaly time window for normal operation modes.

Compared to the RE results, the wind speed sensor is indicated much clearer as the cause for the anomalies found. Both in the heat map as well as the ARCANA importances the wind speed sensor is significantly more anomalous than the other features. The ARCANA importance for the this feature is over 80%, whereas the other features have an importance less than 10%. For the scaled REs this difference was much less prominent: the scaled RE for the wind speed measurements is only slightly higher than that of the feature with the second highest RE, the average generator bearing temperature.

5. Application of ARCANA on offshore WT SCADA data

In this Section ARCANA is applied on offshore WT SCADA data. The two case studies discussed in this Section are from February 2020 and May 2020. The method and the AE set up is described in the next section, followed by an overview of the data set used.

5.1. Method

This Section combines the AE and ARCANA methods described in Sections 3 and 4 and applies it to data of an offshore wind farm. The AE architecture is the same as in Section 3, except for the number of neurons in each layer, since more input features are available (about 250 instead of 55). In this case the three hidden layers have 1350, 50 and 1350 neurons. Amongst various tests of different architectures with three hidden layers and less and different hyperparameter configurations, the three hidden layers with its given number of neurons showed best results. In comparison to 3.1, no further details can be given in this Section due to the confidentiality of the data set.

A model is trained for each WT, therefore not always the same number of sensor measurements are available. Usually a number of around 250 different sensor measurements is used, depending on data availability and data gaps.

The anomaly threshold is calibrated by setting FDR to 0.8 to have a more sensitive behaviour of the AE [5].

5.2. Data set

The characteristics of the data set used in this Section is the same as described in Lutz et al. [5]. The WTs are located in the same wind farm in the German North Sea and are of the same turbine type. The operational data consists of ten-minute-average values of around 250 sensor measurements of the WT. Examples of these measurements are the power production, wind speed and temperature measurements. Also, some measurements represent counter values: The number of cable windings or the energy production since commissioning of the WT. For anomaly detection, we only use measurements, counter values are dropped.

There are 15 different OP-modes possible, e.g.: manual stop, production, ready, failure, grid outage. Since the AE is trained to learn normal behaviour, not all of the timestamps are used. Only those timestamps which OP-modes are considered to be normal are further used for training. The normal OP-modes are: ready, start-up, production, shadowing and high wind shutdown. A timestamp is further discarded and considered to be not normal if within a timestamp normal and not normal OP-modes are present.

The further steps of data preparation are the same as described in Section 3.1.1.

5.3. Results

In this Section the results of applying ARCANA on the before mentioned data set is shown. Several anomalies within the period of consideration have been discussed with the wind farm operator. Out of the anomalies, which have found to be technically relevant, two are further outlined in the following.

5.3.1. Case study one

A constant anomaly beginning on February $25th$, 2020 at 04:30 am ranging until February $29th$, 2020 at midnight is detected. The technical cause for the anomaly is a broken wind speed sensor. During this period every timestamp with a normal OP-mode is considered as an anomaly. For the period of constant anomalies the REs and ARCANA are calculated.

The right side of Fig. 8 shows the average scaled REs for each sensor measurement as vertical bars. The average scaled REs are sorted in descending order. Several sensor measurements were constant during the anomaly with a standard deviation close to zero. This results in a extremely high scaled RE (see Eq. (3)). The highest scaled RE is calculated for wind speed measurement one.

In the left part of Fig. 8 the result of ARCANA within the same time window is seen. With an importance of 45% the wind speed measurement is seen at first, similar as the right part of Fig. 8. The second most

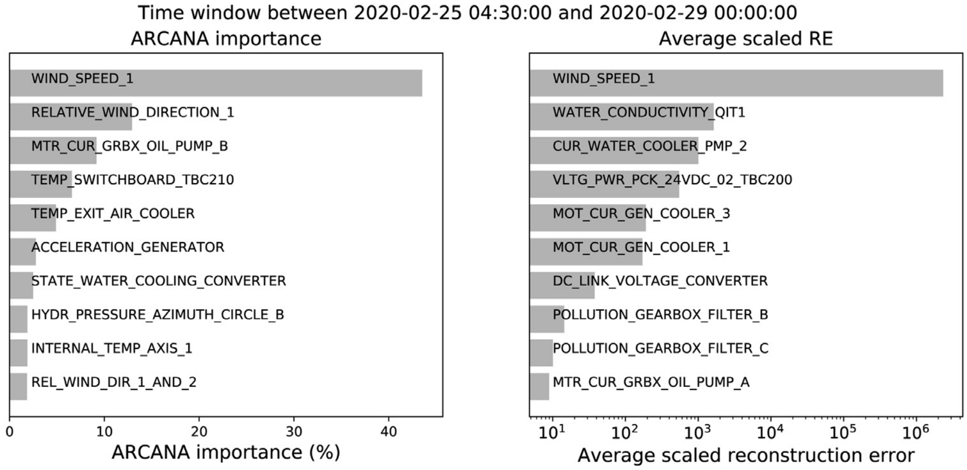


Fig. 8. ARCANA importances (left) and scaled REs (right) for case study one.

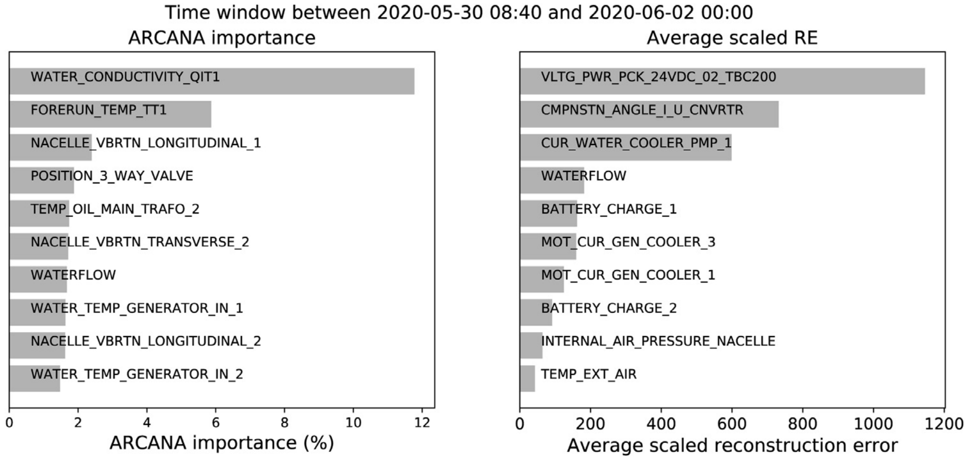


Fig. 9. ARCANA importances (left) and scaled REs (right) for case study two.

important sensor measurement with an importance of 15% is the relative wind direction. The importance can be interpreted as the contribution of that sensor to the anomaly time window.

The scales of the RE are highly deviating and an interpretation is not very intuitive. Also, features that are not technically related to the wind speed measurement show a high RE, e.g. the voltage converter link. ARCANA, however, helps to understand a possible underlying technical consequence that caused the anomaly by pointing out one or more relevant sensors. In this instance the wind speed measurement and relative wind direction.

5.3.2. *Case study two*

In the second case study a change of the converter water conductance is observed. A subcomponent of the converter was retrofitted. As a consequence the conductance changed and this change is detected as an anomaly. The anomaly time window started on May 30*th* at 8:40 am and lasted until June 2*nd* at midnight.

By only using the RE to interpret the anomaly, one assumes that the voltage of the power pack and possibly a converter sensor measurement behave abnormally (see right part of Fig. 9). However this is not the root cause for the anomaly.

By using ARCANA the anomaly can be investigated. The left part of Fig. 9 presents the ARCANA importance of the sensor measurements.

The highest contribution for the anomaly time window is given by the sensor measurement of the water conductivity with an ARCANA importance of around 12%. The water conductivity directly influences the converter water conductance. A technical interpretation with ARCANA is easier, since less time is needed to investigate the source of the abnormal WT state. The second highest contribution for the anomaly time window is given by the forerun temperature. An underlying cause for this contribution is not known to the authors.

Anomalies are often detected after performing service measures at the WT. E.g. a retrofit or oil refill. Usually, the state of the WT after those measures is not critical. By using ARCANA it can be understood that such anomalies are a consequence of the aforementioned measures. In such cases a retraining of the AE model is necessary to learn the new behaviour.

6. Discussion and outlook

We developed ARCANA and applied it on anomaly time windows that have been detected with the usage of an AE. Yet, not all anomaly time windows are a deviating state from the WT design specification or an indication for a possible upcoming failure. Some of those periods are normal operational behaviour but simply not covered in the training data set and thus interpreted as an anomaly, e.g. high wind gusts. We assume that anomalies are detected correctly to apply ARCANA. Furthermore we assume that abnormal WT-states can be seen in the SCADA data.

To calibrate the anomaly score threshold, the FDR method was used. Nevertheless, different threshold calibration methods need to be compared against each other. Again, this is part of future work to improve the anomaly detection method.

ARCANA eases the identification and the interpretation of the root cause of an anomaly by pointing out the feature contributions to that anomaly. Domain knowledge is however still needed to interpret the results and determine the underlying failure or changed environmental conditions. An application in other domains is not tested but with sufficient data available possible to implement. With more instances and input features available AE models are expected to perform better. The amount of instances and input features sufficient to train a model depends on the use case at hand.

An adaptation of the ARCANA method on different domain data with similar characteristics, e.g. sensor data of machinery with a sufficient amount of samples, is expected to be applicable too. Up to now, the authors have tested ARCANA on the two data sets (Sections 5.2 and 3.2) described. In this case, only the AE would need to be optimised for the data used, i.e. testing of different architectures and hyperparameter tuning. No other limits of using different data sets as such described beforehand are expected to be seen.

Also the authors did not compare the results of ARCANA to the results of other explainable ML models on the same data set. Future research should be made to compare not yet related approaches, e.g. ARCANA to a publicly available baseline model on a publicly available data set.

In this study a simple AE model is used. An extension of the ARCANA algorithm for ensemble models is expected to yield better predictive capabilities, as ensemble models almost always perform better than a single model. This should be part of future research. Additionally, most failures can be explained in more than one way, due to correlations between sensor measurements. An extension of the algorithm to find other possible solutions for the ARCANA bias vector may provide a more complete explanation of detected anomalies. This can for example be achieved by generalising the ARCANA idea to a Bayesian approach. Minimising the ARCANA loss function is similar to lasso or ridge regression. Both can be seen as maximum a-posteriori estimates of Bayesian regression. Multiple explanations of a failure can then potentially be linked to different modes of the posterior.

The proposed methods are developed using Python 3.6.8 [32]. To implement the AE keras 2.2.4, Scipy 1.3.0, Tensorflow 1.14.0 and scikit-learn 0.21.2 were used. Data handling is done with cx_oracle 7.2.0, sqlalchemy 1.3.13, numpy 1.16.4 and pandas 0.24.2. Hyperparameter tuning is done by using the Neural Network Intelligence Toolkit [33]

7. Summary

To identify possible root causes of anomalies detected by an AE, a novel method is introduced: ARCANA. The proposed method is compared to anomaly interpretation based on the AE output, which uses the RE of each input feature. Anomaly interpretation based on the REs is surprisingly unintuitive, since a deviating feature can cause large REs in many other features as well. By using ARCANA the number of features with a significant contribution to the anomaly can be highlighted. Additionally, the feature importances for a specific anomaly, or an anomaly time window, can be determined. The possible technical root cause is therefore easier to determine.

This is shown on an open data set of WT SCADA data where an artificial error was added onto the measurements of the wind speed sensor. Using the RE shows that the wind speed is deviating from a normal state, however this behaviour is seen in other sensor measurements as well. We show that ARCANA points out the wind speed sensor correctly and that its feature importance is significantly higher than the importances of other features. It is obvious without having to look at the individual feature time series that this feature is causing the anomaly.

The same method is applied on SCADA data from offshore WTs. Two case studies are discussed, demonstrating the technical relevance of the ARCANA method. First the contribution of a broken wind speed measurement to an anomaly time window is clearly shown by using ARCANA. Second, a rise of the conductance can be seen with ARCANA, which could not be shown by using the RE of the AE.

ARCANA can provide human interpretable output for AEs. We have shown its technical relevance for WT SCADA data, for which it helps to interpret anomalies and to understand the underlying causes.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

The development of ARCANA was funded by the German Federal Ministry for Economic Affairs and Energy through the research project ModernWindABS under grant number: 0324128. The publication of this work was funded by the Hessian Ministry of Higher Education, Research, Science and the Arts through the K-ES project under reference number: 511/17.001.

References

[1] Lei X, Sandborn P, Bakhshi R, Kashani-Pour A, Goudarzi N. PHM based predictive maintenance optimization for offshore wind farms. In: Proceedings of the IEEE conference on prognostics and health management (PHM); 2015. p. 1–8. doi:10.1109/ICPHM.2015.7245027.

[2] Stetco A, Dinmohammadi F, Zhao X, Robu V, Flynn D, Barnes M, et al. Machine learning methods for wind turbine condition monitoring: a review. Renew Energy 2019;133:620–35. doi:10.1016/j.renene.2018.10.047.

[3] Helbing G, Ritter M. Deep learning for fault detection in wind turbines. Renew Sustain Energy Rev 2018;98:189–98. doi:10.1016/j.rser.2018.09.012.

[4] Chalapathy R., Chawla S.. Deep learning for anomaly detection: a survey2019; arxiv. 1901.03407v2

[5] Lutz M-A, Vogt S, Berkhout V, Faulstich S, Dienst S, Steinmetz U, et al. Evaluation of anomaly detection of an autoencoder based on maintenace information and SCADA-data. Energies 2020;13(5):1063. doi:10.3390/en13051063.

[6] Renström N, Bangalore P, Highcock E. System-wide anomaly detection in wind turbines using deep autoencoders. Renew Energy 2020;157:647–59. doi:10.1016/j.renene.2020.04.148.

[7] Wu X, Jiang G, Wang X, Xie P, Li X. A multi-level-denoising autoencoder approach for wind turbine fault detection. IEEE Access 2019;7:59376–87. doi:10.1109/ACCESS.2019.2914731.

[8] Jiang G, Xie P, He H, Yan J. Wind turbine fault detection using a denoising autoencoder with temporal information. IEEE/ASME Trans Mechatron 2018;23(1):89–100. doi:10.1109/TMECH.2017.2759301.

[9] Zhao H, Liu H, Hu W, Yan X. Anomaly detection and fault analysis of wind turbine components based on deep learning network. Renew Energy 2018;127:825–34. doi:10.1016/j.renene.2018.05.024.

[10] Wang L, Zhang Z, Xu J, Liu R. Wind turbine blade breakage monitoring with deep autoencoders. IEEE Trans Smart Grid 2018;9(4):2824–33. doi:10.1109/TSG.2016.2621135.

[11] Yuan B., Wang C., Luo C., Jiang F., Long M., Yu P.S., et al. Waveletae: a wavelet-enhanced autoencoder for wind turbine blade icing detection2019; arxiv. 1902.05625v2

[12] Vilone G., Longo L.. Explainable artificial intelligence: a systematic review. (2020) arxiv. 2006.00093v4

[13] Belle V., Papantonis I.. Principles and practice of explainable machine learning. arxiv. 2009.11698v1

[14] Chatterjee J, Dethlefs N. Deep learning with knowledge transfer for explainable anomaly prediction in wind turbines. Wind Energy 2020;23(8):1693–710. doi:10.1002/we.2510.

[15] Marti-Puig P, Blanco-M A, Cárdenas J, Cusidó J, Solé-Casals J. Feature selection algorithms for wind turbine failure prediction. Energies 2019;12(3):453. doi:10.3390/en12030453.

[16] Zhang D, Qian L, Mao B, Huang C, Huang B, Si Y. A data-driven design for fault detection of wind turbines using random forests and xgboost. IEEE Access 2018;6:21020–31. doi:10.1109/ACCESS.2018.2818678.

[17] Ribeiro MT, Singh S, Guestrin C. Why should i trust you?. In: Krishnapuram B, Shah M, Smola A, Aggarwal C, Shen D, Rastogi R, editors. Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining. New York, NY, USA: Association for Computing Machinery; 2016. p. 1135–44. ISBN 9781450342322. doi:10.1145/2939672.2939778.

[18] Shrikumar A, Greenside P, Kundaje A. Learning important features through propagating activation differences. In: Proceedings of the 34th international conference on machine learning, PMLR 70. JMLR.org; 2017. p. 145–3153. http://proceedings.mlr.press/v70/shrikumar17a.html

[19] Lundberg SM, Lee S-I. A unified approach to interpreting model predictions. In: I Guyon, U V Luxburg, S Bengio, H Wallach, R Fergus, S Vishwanathan, et al., editors. Advances in neural information processing systems 30. Curran Associates, Inc; 2017. p. 4765–74.

[20] Antwarg L., Miller R.M., Shapira B., Rokach L.. Explaining anomalies detected by autoencoders using shap2020; arxiv. 1903.02407v2

[21] Tagawa T, Tadokoro Y, Yairi T. Structured denoising autoencoder for fault detection and analysis. In: Dinh Phung, Hang Li, editors. Proceedings of the sixth Asian conference on machine learning. Proceedings of Machine Learning Research, 39. Nha Trang City, Vietnam: PMLR; 2015. p. 96–111. http://proceedings.mlr.press/v39/tagawa14.html

[22] Nguyen Q.P., Lim K.W., Divakaran D.M., Low K.H., Chan M.C.. Gee: a gradient-based explainable variational autoencoder for network anomaly detection. In: Proceedings of the IEEE conference on communications and network security (CNS). IEEE. ISBN 978-1-5386-7117-7; 10.06.2019 - 12.06.2019, p. 91–99.. 10.1109/CNS.2019.8802833

[23] Ikeda Y., Ishibashi K., Nakano Y., Watanabe K., Kawahara R.. Anomaly detection and interpretation using multimodal autoencoder and sparse optimization2018; arxiv. 1812.07136v1

[24] The Canadian Institute for Cybersecurity. NSL-KDD dataset. 2009. https://www.unb.ca/cic/datasets/nsl.html.

[25] Chollet F., et al. Keras. https://keras.io; 2015.

[26] Abadi M., Agarwal A., Barham P., Brevdo E., Chen Z., Citro C., et al. TensorFlow: large-scale machine learning on heterogeneous systems. 2015. Software available from tensorflow.org; https://www.tensorflow.org/.

[27] Rousseeuw PJ, van Driessen K. A fast algorithm for the minimum covariance determinant estimator. Technometrics 1999;41(3):212–23. doi:10.1080/00401706.1999.10485670.

[28] Vogt S., Berkhout V., Lutz A., Zhou Q.. Deep learning based failure prediction in wind turbines using SCADA data2019;.

[29] Diederik P. Kingma, Jimmy Ba. Adam: a method for stochastic optimization. In: Yoshua Bengio, Yann LeCun, editors. Proceedings of the 3rd international conference on learning representations; 2015. 1412.6980v9

[30] EDP Inovação. EDPR wind farm open data: wind turbine SCADA signals and historical failure logbook from 2016 and 2017. 2018. https://opendata.edp.com/pages/Windfarms/.

[31] International Electrotechnical Commission. IEC 61400-1: wind turbines - part 1: design requirements. 2019.

[32] Python Software Foundation. python. 2021. https://www.python.org/.

[33] Microsoft. Neural network intelligence. 2021. https://github.com/microsoft/nni.