ITAT 2016 Proceedings, CEUR Workshop Proceedings Vol. 1649, pp. 123–130
http://ceur-ws.org/Vol-1649, Series ISSN 1613-0073, © 2016 T. Kuzin, T. Borovička

Early Failure Detection for Predictive Maintenance of Sensor Parts

Tomáš Kuzin, Tomáš Borovička

Faculty of Information Technology,
Czech Technical University in Prague,
Prague, The Czech Repubic
kuzintom@fit.cvut.cz,
tomas.borovicka@fit.cvut.cz

Abstract: Maintenance of a sensor part typically means renewal of the sensor in regular intervals or replacing the malfunctioning sensor. However optimal timing of the replacement can reduce maintenance costs. The aim of this article is to suggest a predictive maintenance strategy for sensors using condition monitoring and early failure detection based on their own collected measurements.

Three different approaches that deal with early failure detection of sensor parts are introduced 1) approach based on feature extraction and status classification, 2) approach based on time series modeling and 3) approach based on anomaly detection using autoencoders. All methods were illustrated on real-world data and were proven to be applicable for condition monitoring.

1 Introduction

In the last decade the amount of used sensors across all sectors has significantly raised. This is important and a still continuing trend.

In the classical concept, predictive maintenance takes place when the maintained asset is expensive or important for key business processes. In other words when proper utilization of the machinery has important economic or safety consequences. This is not the characteristic case of sensor parts which are usually cheap and play a minor role. For such assets maintenance typically means simple replacement and reactive maintenance strategy would be the most common choice. However, machines become more and more dependent on sensor parts and that brings new challenges in their maintenance. Proper timing of replacement has direct influence on maintenance expenses. Especially in cases where other processes depend on the sensor readings and the sensor failure or malfunction may stop the operation or cause collateral loses of a machinery.

In case of sensors "classical" condition monitoring scheme utilizing properly chosen set of external sensors makes no sense. On the other hand sensors themselves provide on-line measurements during their whole operational service. These data may be exploited to estimate the current state of the measuring device. Therefore applying smarter maintenance strategy for sensor parts makes perfect sense and may introduce significant savings.

This article deals with the possibilities of smarter maintenance strategies for sensor parts. The main idea is to apply machine learning techniques in order to monitor the current condition or predict failure of sensors based on their own measurements and propose an optimal time for their replacement in order to avoid failures.

2 Related Work

Several articles and works on "classical" predictive maintenance and condition monitoring [1, 2] were published in the literature. Predictive maintenance strategy is usually a rule-based maintenance grounded on on-line condition monitoring, which relies on an appropriately chosen set of external sensors. The proper sensor set plays the key role [2]. Unfortunately none of these techniques are useful if it is needed to monitor the state of sensors themselves. Moreover, many published works base their approaches on sensor networks, where malfunction of one sensor can be identified utilizing measurements of other sensors in the network. However, this paper focuses on "standalone" sensors where no more devices sensing the same or correlated phenomena are available. Thus these approaches use only measurements of the sensor itself. Since there are not many available publications for this case, further review is focused on categorization of faults and fault detection techniques of both the sensors and sensor networks.

Sensors provide a huge amount of information about observed phenomena. However, to make meaningful conclusions, the quality of the data has to be ensured. Sensors alone can malfunction and that can distort an image of the phenomena. Most of the methods follow a common framework, characterize the normal behavior of sensor readings, identify significant deviations and mark them as faults.

In case of sensor networks the most frequent types of faults have been described and categorized by Ni, K. et al[3]. They describe two distinct approaches to deal with faults. The first is a data-centric view which examines the data collected by a given sensor and describes fault models based on data features. In contrary there is a system-centric view which examines physical malfunctions of a sensor and how those may manifest themselves in the resulting data. According to Ni et. al. these two views are related to one another and every fault can be mapped between these two. The important fault categories discussed in [3] are summarized in Table 1. In this article the focus is on the data centric point of view.

Sharma, A.B. et al.[4] loosely follow on the work of Ni et al. and propose specific algorithms for fault detection.

They focus only on a subset of fault types examined in [3] and summarized in Table 1.

Table 1: Taxonomy of Faults described by Ni et al.[3].

Data-centric point of view

Fault	Definition
Outlier	Isolated data point or sensor unexpectedly distant from models.
Spike	Multiple data points with a much greater than expected rate of change.
“Stuck-at”	Sensor values experience zero variation for an unexpected length of time.
High Noise or Variance	Sensor values experience unexpectedly high variation or noise.

Four different classes of approaches for detecting above mentioned faults are discussed.

Rule-based Methods use domain knowledge to develop heuristic constraints that the sensor readings must satisfy. Violations of those constraints imply faults. For above mentioned fault types following simple rules are typically used [4]:

The variance (or the standard deviation) of the sample readings within a window of size w_{size} is computed. If it is above a certain threshold, the samples are corrupted by the noise fault. If the variance is zero the samples are corrupted by the constant fault. In order to detect short noise faults, the data had to be appropriately preprocessed. If the rate of change is above a threshold, it can be assumed that the data were affected by short faults.

The performance of this method strongly depends on parameters w_{size} and the threshold. Parameter setting is not trivial and usually requires domain knowledge of the examined problem.

Estimation-based Methods can be used when a physical phenomena is sensed concurrently by multiple sensors and dependence between sensor measurements can be exploited to generate estimates for the individual sensor measurements. The dependence can be expressed by spatial correlation. Regardless of the cause of the correlation, it can be used to model the normal behavior. The estimation can be done for example by Linear Least-Squares Estimation. This method is most suitable for cases when the phenomena is sensed by almost identical sensors. As an example one can imagine multiple barometric altimeters on a single aircraft. In this case there is a strong presumption that the values are strongly correlated.

Time-series-based Methods utilize the fact, that measurements of a sensor are not random and therefore contain some kind of regular patterns. This patterns can be described through autocorrelations in measurements collected by a single sensor. These can be used to create a regressive model of sensed phenomena. A sensor measurement can be than compared against its predicted value to determine if it is faulty.

Advantage is that this approach is more general than classification and can be used even if there are no labeled data available nor multiple strongly correlated sensors.

Learning-based Methods use training data to infer model of "normal" sensor behavior. If the "normal" sensor behavior and the effects of sensor faults are well understood, learning-based methods may be suitable to detect and classify sensor faults. In [4] authors successfully use Hidden Markov Models to construct a model of sensor measurements. The main advantage of learning based methods is that they can simultaneously detect and classify faults.

3 Preliminaries

3.1 Classification

In the terminology of machine learning, classification is considered an instance of supervised learning, i.e. machine learning technique where a training set of correctly identified observations is available [5]. The main goal of classification is assigning a new observation X to one from a finite set of categories with the use of the training data set containing instances whose category membership is known.

Every instance of the input dataset is a vector $X = (x_1, x_2, \ldots, x_d)$ typically called feature vector, where d is the number of features ($0 < i <= d$) and x_i is the value of the $i^t h$ feature. Every instance belongs to one of the k classes $C = c_1, c_2, \ldots, c_k$.

The classification process consists of two phases. In the first phase, called learning phase, the training data set with labels is used to build a model. It means that the knowledge from reference data is being extracted and stored in form of a model. In the second phase, the model is used to classify unlabeled data. This phase is often called recall. An algorithm that implements classification is called classifier.

Naive Bayes In machine learning naive Bayes classifiers are a family of probabilistic classifiers based on Bayes theorem. It assumes that a value of a particular feature is independent of a value of any other feature, given the class variable [6]. This assumption is often violated in practice but even though Naive Bayes classifier is still powerful classification techniques.

Learning naive Bayes model proceeds with calculation of probabilities from the training data set. The probability to be estimated is a conditional probability $P(c_j|x_1, ..., x_d)$

for each class c_j when object $X = (x_1, x_2, \dots, x_d)$ is given [7].

Using the Bayes rule

$$P(A \mid B) = \frac{P(A)P(B \mid A)}{P(B)} \tag{1}$$

the posterior probability can be expressed by Equation

$$P(c_j \mid X_1, \dots, X_d) = \frac{P(c_j)P(x_1, \dots, x_d \mid c_j)}{P(x_1, \dots, x_d)}, \tag{2}$$

where

- $P(c_j \mid x_1, \dots, x_d)$ is the posterior probability of class c_j when object $X = (x_1, x_2, \dots, x_d)$ is given.
- $P(c_j)$ is the prior probability of class c_j.
- $P(x_1, \dots, x_d \mid c_j)$ is the posterior probability of an object $X = (x_1, \dots, x_d)$ when class c_j is given. We call this probability likelihood.
- $P(x_1, \dots, x_d)$ is the prior probability of an object $X = (x_1, x_2, \dots, x_d)$.

The resulting model is represented by prior probabilities of each class and likelihood probabilities for each combination of class and feature. The likelihoods are usually represented by a mean and variance of normal distribution estimated from the training set.

The recall of naive Bayes algorithm is done by looking up the prior and likelihood probabilities which belong to input data and calculating posterior probabilities for each class. Thanks to the assumption of strong conditional independence between all features conditioned by the class, the likelihood can be calculated as follows.

$$P(x_1, \dots, x_d \mid c_j) = \prod_{i=1}^{n} P(x_i \mid c_j) \tag{3}$$

The resulting class is determined by the highest posterior probability.

3.2 Time Series Modeling

Time series is a series of observations of a process or an event in equal time intervals. It is called time series, because the observations are usually taken with respect to time. This is however not necessity, because the observations may be taken with respect to space as well [8].

Modeling techniques try to find a model which describes the series, i.e. a model capable to generate identical series. The model may help to better understand the underlying phenomena or serve as forecasting tool to predict future values of the series.

Stochastic models like ARIMA assume that the time series consist of regular pattern manifesting the underlying phenomena and a random noise.

The ARIMA Model ARIMA (autoregressive integrated moving average model) is a general time series model. It combines two independent models, autoregressive (AR) and moving-average (MA). They are combined in a single equation (Equation 4). By convention the AR terms are added and the MA terms are subtracted.

$$x_t = C + \varphi_1 \cdot x_{t-1} + \cdots + \varphi_p \cdot x_{t-p} - \theta_1 \cdot \varepsilon_{t-1} - \cdots - \theta_q \cdot \varepsilon_{t-q} \tag{4}$$

where

- x_i is i-th element of the series,
- C is a constant,
- φ_1, φ_2 are parameters of the autoregressive model,
- ε_i is random error component of i-th member of the series.
- θ_1, θ_2 are parameters of the moving average model.

ARIMA models are extensively examined in literature. For more information the reader is reffered to [9] or [10].

3.3 Artificial Neural Networks

Artificial neural network is an information processing paradigm inspired by biological nervous systems. It is composed of a large number of highly interconnected processing units (neurons) working in unity to solve a specific problems.

A neuron is a simplistic model of a biological neural cell. Each neuron has one or more inputs and produces single output. The inputs simulate the stimuli signals that the neuron gets from other neurons, while the output simulates the response signal which the neuron generates.

The biological neuron fires (i.e generates the response signal) only if the gathered stimuli signals exceed a certain threshold. In other word the neuron fires only if the $stimuli - treshold > 0$. In the context of ANNs the term bias b is used instead of "threshold"[1].

The artificial equivalent to gathered stimuli signals is called inner potential (ξ) and typically is defined as a weighted sum of the input signals plus the bias. Each input (x_j) is multiplied by a specific real number w_j called the weight. These weights are parameters of each neuron. The calculation of inner potential is summarized in Equation 5.

$$\xi = \sum_{all\, j} w_j * x_j + b = W \cdot X + b \tag{5}$$

The actual output is obtained by applying activation function $\varphi(\cdot)$ on the gathered inner potential. There can be used variety of activation functions. Very popular for

[1]Due the conventions $bias = (-1 \cdot threshold)$.

its properties is sigmoid function, where the output of the neuron y is given by formula in Equation 6.

$$y = \varphi(\xi) = \frac{1}{1+e^{-\xi}} \qquad (6)$$

For more complex tasks like anomaly detection a single neuron is not powerful enough and therefore more complex structures are introduced. A neural network is a group of neurons connected together. Connecting neurons to form a ANN can be done in various ways.

Networks where the neurons are arranged in separate layers and the output from one layer is used as an input to the next layer are called feed-forward networks. This means there are no loops in the network and information is always fed forward, never fed back.

ANNs, like their biological artworks, learn by example. Therefore in order to train a neural network a set of input examples with known expected responses is necessary. Classical method of training ANNs is called "backpropagation" which is an abbreviation for "backward propagation of errors".

Typical goal in a training of neural networks is to find weights $W = (w_1, \ldots, w_k)$ and biases $B = (b_1, \ldots, b_l)$ which minimize the error or cost function $C(W,B)$ over all instances in the training set.

More specific information about different ANN types can be found in literature [11, 12].

Autoencoder is a specific type of feed-forward neural network, with an input layer, an output layer and one or more hidden layers. The main properties of an autoencoder are, that the output layer has the same number of neurones as the input layer and instead of being trained to predict some target value Y given inputs X, autoencoders are trained to reconstruct their own inputs X'.

Especially interesting are autoencoders, where hidden layers have less nodes than input/output layer. Such a network is forced to comprehend nonlinear, reduced representation of the original data.

Such a autoencoder network can have a variety of uses. They can serve for non linear dimensionality reduction, data compression or to learn generative model of the data[13].

4 Approach

Influenced by related work reviewed in the Section 2, three different approaches to deal with condition monitoring of sensor parts are introduced. Each approach is based on a different principle; the first approach is based on feature extraction and status classification, the second approach is based on time series modeling and the third approach is based on anomaly detection using autoencoders. Approaches are illustrated on data set with measurements from 2000 accelerometers (hereafter referred as sensors). For each sensor the data set contains one time series with minimum of 14 days measurements before the sensor failed. The aim is to label the sensor faulty within two days before the failure. More than two days before the failure the sensor can be considered faultless. All three approaches are described in detail in the following subsections.

4.1 Classification-based approach

The first suggested approach is based on supervised learning, namely classification. Supervised learning techniques require examples with labels to learn from. This approach, therefore, requires information about failures to prepare the labels. If no information about the failures is available and the labels can not be supplied this approach can not be applied.

The sensor readings are in a form of a time-series. Sliding window of N measurements is used to calculate the feature vector for classification. The raw measurements itself can be used directly as a feature vector, however, the dimensionality is then equal to the size of the sliding window multiplied by the number of measured phenomenons. Typically, simple features (such as variance, average, median or slope) or more complex features (e.g. Fourier or wavelet coefficients) are extracted from the sliding window [14, 15, 16]. For on-line condition monitoring the feature vector is extracted from a window aligned with the most current readings. The instance is then classified by pre-trained classifier. If the instance is classified as "failed" the current condition of the sensor is evaluated as faulty.

In order to train a model the labels have to be prepared. To prepare the training dataset historical readings and a set of times related to the failures or generally the events to be detected are used. To obtain faulty instance sliding window is placed over the readings of a failed sensor and aligned with the time of failure. A feature vector is extracted from such a window and marked with label "failed" (i.e. class y=1). For each failure one instance with a label "failed" is obtained. Non-faulty instances can be extracted by sliding the window over the time series of non-failed sensor[2].

However, by using every possible shift unnecessarily large number of instances is obtained. Therefore, non-faulty instances are extracted by placing the window randomly over readings. Extracted feature vectors are marked with label 'ok' (i.e. class y=0). In this case the ratio between classes can be easily controlled. The whole process demonstrates Figure 1.

The number of features is reduced with iterative forward feature selection method. Initially a model is trained with only one feature, in each iteration one feature as added and model is retrained. If the new model performs significantly better than the previous, the feature is kept in the feature vector, otherwise the feature is discarded.

[2]Non-failed sensor is a sensor for which do not exist any record of failure.

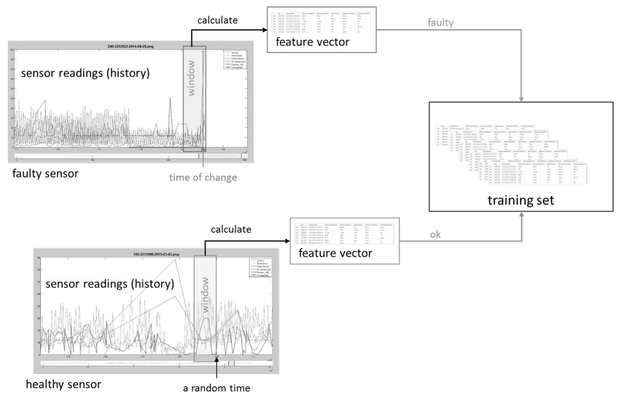

Figure 1: Working scheme of creating the training set.

Classification model is trained with extracted feature vectors to recognize faulty and non-faulty instances. Arbitrary classifier can be used. The aim of this article is to prove the concept that classification can be used for condition monitoring and thus maintenance strategy for sensor parts. Therefore for simplicity and interpretability the Naive Bayes classifier is applied.

In Naive Bayes the instance is typically classified to a class with higher posterior probability. To increase confidence of positive classification the minimal threshold value of posterior probability can be set on the class with failed instances. With this threshold of minimal probability for positive class can be controlled trade-off between sensitivity and specificity of the naive Bayes classifier. With a higher threshold the classifier will be more certain about the prediction, however, it may mark more failures as non-faulty and vice versa.

4.2 Time-series Modeling-based approach

The second approach basically follows the method suggested in [4]. It assumes that malfunction of a sensor is preceded by an abnormal behavior. The working principle basically follows the common framework for anomaly detection. It uses time-series modelling in order to model "normal" sensor behavior.

A regressive model is trained on the historical measurements of a specific sensor and used to generate predictions. The ARIMA model[9, 10] is general regressive model popular in time-series modeling. Especially in cases, when the time-series contains significant regular patterns, which is more or less the case of sensor readings [4]. For that reason the general ARIMA model is used to obtain the predictions.

Predicted values are compared with the actual readings and if the difference is higher than a certain threshold, measurements are marked as faulty.

The working scheme is depicted in Figure 2.

The ARIMA model prescription contains random members, therefore it is a stochastic process. In order to re-

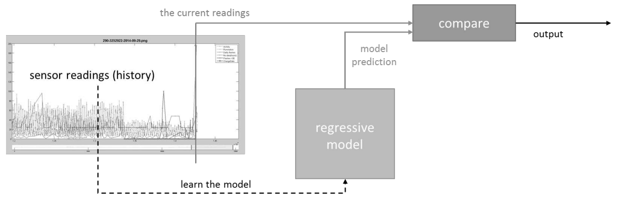

Figure 2: Working scheme of regression-model-based approach

duce random component and get the most precise predictions Monte Carlo principle is typically engaged to generate multiple predictions. The final prediction is obtained as a mean value of k predicted values.

Knowing how the prediction is obtained allows us to create hypothesis about the expected value and construct a confidence interval for the predicted value as shown in Figure 3.

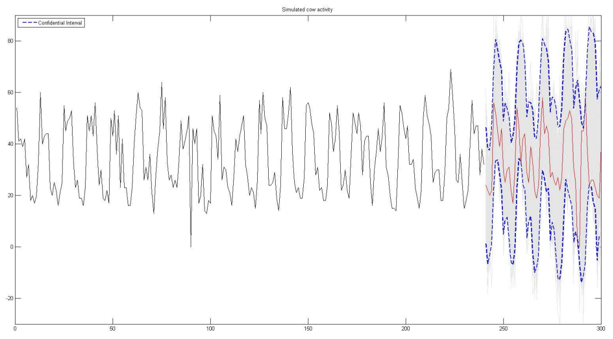

Figure 3: ARIMA model predictions with the confidential interval.

If the actual reading of a sensor is out of the confidence interval of the corresponding predicted value the sensor is marked as ‘faulty’.

4.3 Autoencoder-based approach

The last suggested approach is, similarly to the previous approach, based on an assumption that the failure of a sensor is preceded by its anomalous behavior. In this particular case auto-encoders are utilized to detect anomalies.

Inputs to the autoencoder network are the raw values from a sliding window drawn over historical measurements of the sensor. However, it is also possible to extract different features and use them as inputs of the autoencoder. As a result, this method requires a certain amount of historical data, in order to train an autoencoder network.

The whole working scheme is shown in Figure 4.

The structure of an autoencoder is defined by following parameters: size of the input and output layer, number of hidden layers and number of nodes in the hidden layers.

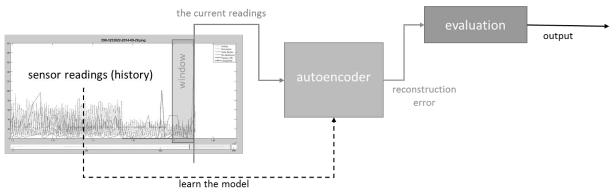

Figure 4: Working scheme of anomaly-detection-based approach.

Since the raw data from the sensor are used as inputs to the autoencoder network, the number of nodes in the input and also the output layer is determined by size of the sliding window. Influenced by [13] the autoencoder has three hidden layers. The number of neurons is related to the number of neurons in the input / output layer. Let n be the number of input respectively output neurons than the hidden layers have $0.75n, 0.5n, 0.75n$ neurons.

The output of an autoencoder itself is not especially interesting. Rather a reconstruction error, defined as mean squared error between the real measurements and output of the autoencoder, is calculated. Let $X = (x_1 \ldots x_n)$ be the input vector of an autoencoder network and $X' = (x'_1 \ldots x'_n)$ is the corresponding output, the reconstruction error is

$$RE(X) = \frac{1}{n} \sum_{1}^{n} (x_i - x'_i)^2$$

If the reconstruction error is higher than a certain threshold τ the current condition of a sensor is marked as "faulty".

The threshold is estimated with a heuristic method. The main idea is to consider the reconstruction error being a random variable. Then the underlying distribution of the random variable can be easily estimated. Having a distribution of the reconstruction error, if the value of the error does not lie in a right-sided (upper) confidence interval with confidence level α it is marked "faulty".

$$P(RE(X) < \tau) = 1 - \alpha$$

Figure 5 demonstrated the histogram which is used in order to estimate the underlying distribution function of the reconstruction error.

5 Experimental Results

5.1 Classification-based Approach

Having labeled data, performance of a classifier can be easily measured. TPR (true positive rate) is defined as number of detected failures to the number of all failures in a given dataset. FPR (false positive rate) is defined as the number of positively identified to the number of all negative samples in a dataset. The acquired results are presented by the ROC curve showed in Figure 6. In a ROC

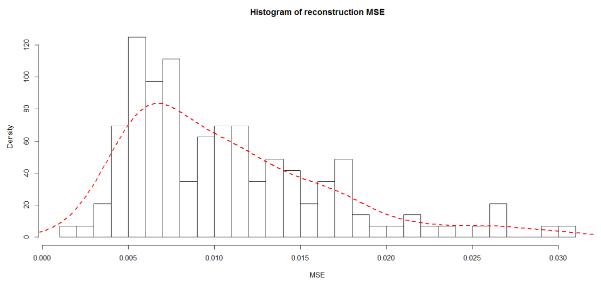

Figure 5: Histogram of reconstruction errors.

curve the TPR (i.e. True Positive Rate or Sensitivity) is plotted in function of the FPR (False Positive Rate or (1-Specificity)) for different setting of model's parameters.

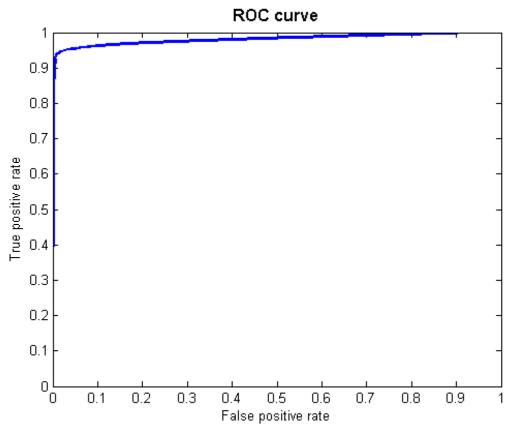

Figure 6: Classification-based approach - ROC curve.

5.2 Time-series Modeling-based Approach

In order to evaluate this approach on predicting failures the method is evaluated as a binary classifier.

A window of a size M is placed before the time of a failure and if an anomaly is within the window, the failure is considered as detected. If an anomaly is detected outside of this window it is considered as false positive detection.

As presented in the section 4.2 this method marks as anomalies all the moments, where the actual reading is not within the confidence interval.

The level of significance α can be set explicitly, and its effect can be examined. In Figure 7 are shown detected anomalies for $\alpha = 0.0015$. The red segments mark the times of failures.

The experiment is repeated multiple times for different α. The results are presented by the ROC curve showed in Figure 8. Each point on the ROC curve represents a TPR/FPR pair corresponding to a particular value of α. It demonstrates how the sensitivity versus specificity can be controlled by choosing the α.

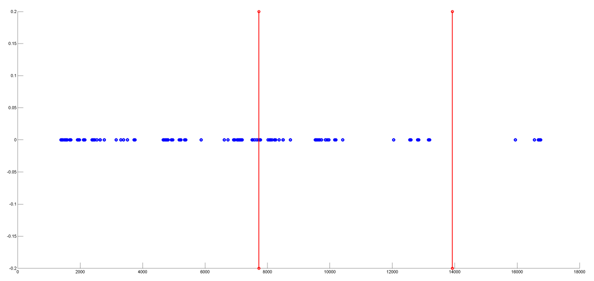

Figure 7: Time-series modeling based approach - detected failures.

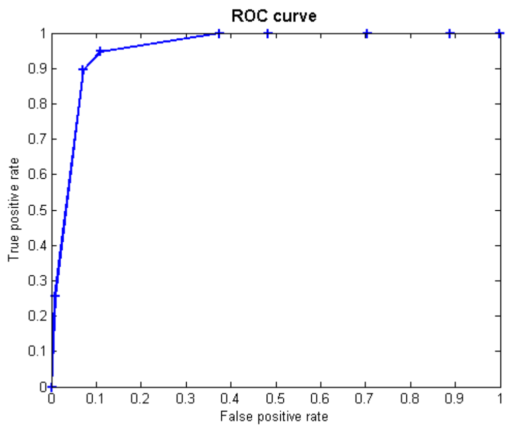

Figure 8: TS-modeling-detection-based approach - ROC curve.

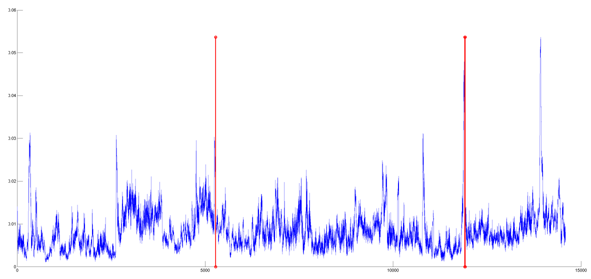

Figure 9: Anomaly-detection-based approach - detected failures.

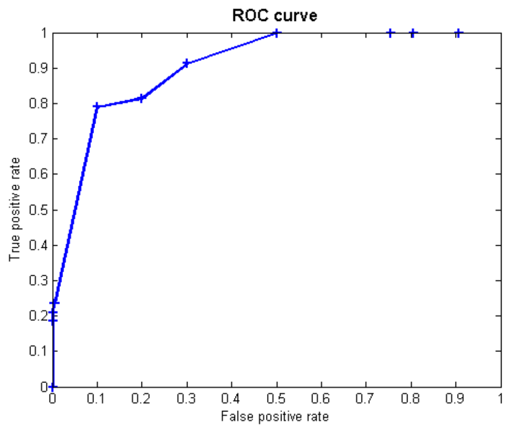

Figure 10: Anomaly-detection-based approach - ROC curve.

5.3 Autoencoder-based Approach

In order to evaluate the autoencoder-based method, the same procedure as in the case of time-series modeling based approach, is used. In Figure 9 is shown the resulting series of the reconstruction errors. The red-marked points are the moments of failure(i.e. the event one intend to predict). It is visible that the time of failure is preceded by significant raise of reconstruction error. However, there are also other anomalous moments(peaks in the series of reconstruction errors), that are not related to the incoming failure of the sensor. Having the domain knowledge of the sensor operation, those can be easily explained, since they are related to the observed phenomena.

The ROC curve in Figure 10 presents the results of this approach. Sensitivity and specificity trade-off is controlled by the *level of significance* described in the the Section 4.3.

6 Conclusion

Three different approaches to deal with the condition monitoring and predictive maintenance of sensors have been described and illustrated on real-world data. All those approaches are chosen with a regard to be general and thus applicable with various sensor devices. All three methods exploit different principles and hence have different assumptions and requirements.

Classification based approach utilizes labels if available. If not this approach is not applicable. The other two approaches are more general since they do not require any meta-data and work just with the sensor measurements. However, both assume that the failure is preceded by anomalous behavior. The time series modeling approach exploits the fact that sensors measurements are in a form of time-series and often contain regular patterns, which manifest themselves in a form of autocorrelations. Therefore they it can be described by a model. The autoencoder-based approach contrary to the time-series modeling does not model the "normal" behavior.

All methods were able to detect failures before they occurred and thus proved to be applicable for condition monitoring and utilized for predictive maintenance of sensor parts. Further more, all the approaches can be parametrize to find an ideal trade-off between sensitivity and specificity of the prediction. The best results has the approach based on classification. This can be expected considering the fact that, unlike the other two approaches, it uses additional meta-data (labels) about the sensor failures.

References

[1] Inc., M. A. Common maintenance strategies. 2015, [Online; accessed 17-January-2016]. Available from: `https://www.maintenanceassistant.com/`

[2] Kennedy, S. New tools for PdM. 2006, [Online; accessed 17-February-2016]. Available from: `http://www.plantservices.com/articles/2006/072/`

[3] Ni, K.; Ramantahan, N.; Nabil, M.; et al. Sensor Network Data Fault Types. *ACM Transactions on Sensor Networks*, volume 5, no. 3, May 2009.

[4] Sharma, A. B.; Golubchi, L.; Govindan, R. Sensor Faults: Detection Methods and Prevalence in Real-World Datasets. *ACM Transactions on Sensor Networks*, volume 6, no. 3, June 2010.

[5] Alpaydin, E. *Introduction to machine learning*. MIT Press, second edition, 2010, ISBN 978-0-262-01243-0.

[6] Russell, S.; Norvig, P. *Artificial Intelligence: A Modern Approach*. Prentice Hall, second edition, 2003, ISBN 978-0137903955.

[7] StatSoft. Naive Bayes Classifier. 2016, [Online; accessed 19-February-2016]. Available from: `http://www.statsoft.com/textbook/naive-bayes-classifier`

[8] Vu, K. M. *Optimal Discrete Control Theory: The Rational Function Structure Model*. Ottawa: AuLac Technologies, 2007, ISBN 978-0-9783996-0-3, 51–99 pp.

[9] Nau, R. Lecture notes on forecasting. 2014, [Online; accessed 19-February-2016]. Available from: `http://people.duke.edu/~rnau/Slides_on_ARIMA_models--Robert_Nau.pdf`

[10] Lu, Y.; Simaan, M. A. Automated Box–Jenkins forecasting modelling. *Elsevier Automation in Construction 18*, November 2008: pp. 547–558.

[11] Nielsen, M. *Neural Networks and Deep Learning*. Determination Press, 2015. Available from: `http://neuralnetworksanddeeplearning.com/index.htm`

[12] Goodfellow, I.; Bengio, Y.; Courville, A. Deep Learning, 2016, book in preparation for MIT Press. Available from: `http://www.deeplearningbook.org`

[13] Candel, A.; Lanford, J.; LeDell, E.; et al. Deep Learning with H2O. 2015, third Edition. Available from: `https://h2o.gitbooks.io/deep-learning/`

[14] Fu, T.-c. A review on time series data mining. *Engineering Applications of Artificial Intelligence*, volume 24, no. 1, 2011: pp. 164–181.

[15] Chen, Y.; Nascimento, M. A.; Ooi, B. C.; et al. Spade: On shape-based pattern detection in streaming time series. In *Data Engineering, 2007. ICDE 2007. IEEE 23rd International Conference on*, IEEE, 2007, pp. 786–795.

[16] Xing, Z.; Pei, J.; Philip, S. Y.; et al. Extracting Interpretable Features for Early Classification on Time Series. In *SDM*, volume 11, SIAM, 2011, pp. 247–258.