Detecting Anomalous Behavior Towards Predictive Maintenance

Athanasios Naskos[1(✉)], Anastasios Gounaris[2], Ifigeneia Metaxa[1], and Daniel Köchling[3]

[1] ATLANTIS Engineering Ltd., Thessaloniki, Greece
{naskos,metaxa}@abe.gr
[2] Aristotle University of Thessaloniki, Thessaloniki, Greece
gounaria@csd.auth.gr
[3] BENTELER GmbH, Paderborn, Germany
daniel.koechling@benteler.com

Abstract. A key Industry 4.0 element is predictive maintenance, which leverages machine learning, IoT and big data applications to ensure that the required equipment is fully functional at all times. In this work, we present a case study of smart maintenance in a real-world setting. The rationale is to depart from model-based and simple rule-based techniques and adopt an approach, which detects anomalous events in an unsupervised manner. Further, we explore how incorporation of domain knowledge can assist the unsupervised anomaly detection process and we discuss practical issues.

Keywords: Anomaly detection · Predictive maintenance · Industry 4.0

1 Introduction

Predictive Maintenance (PdM) is considered a key task in Industry 4.0 with a view to decreasing, if not eliminating, machinery downtime and operational costs [1]. Modern PdM heavily relies on machine learning, e.g., [10,11,13], where the key concept is to intensively process event logs and then train models to identify failure patterns well in advance.

In this work, we advocate a complementary approach, where we employ unsupervised machine learning, and more specifically anomaly detection [2]. We show that, in addition to devising predictive models, we can continuously monitor the incoming data and apply state-of-the-art algorithms for streaming outlier detection. The outliers reported by such a process can be safely regarded as early signs of failure and thus become part of an advanced PdM solution. The key strength of our approach is that it does not rely on representative event logs and model training.

© Springer Nature Switzerland AG 2019
H. A. Proper and J. Stirna (Eds.): CAiSE 2019 Workshops, LNBIP 349, pp. 73–82, 2019.
https://doi.org/10.1007/978-3-030-20948-3_7

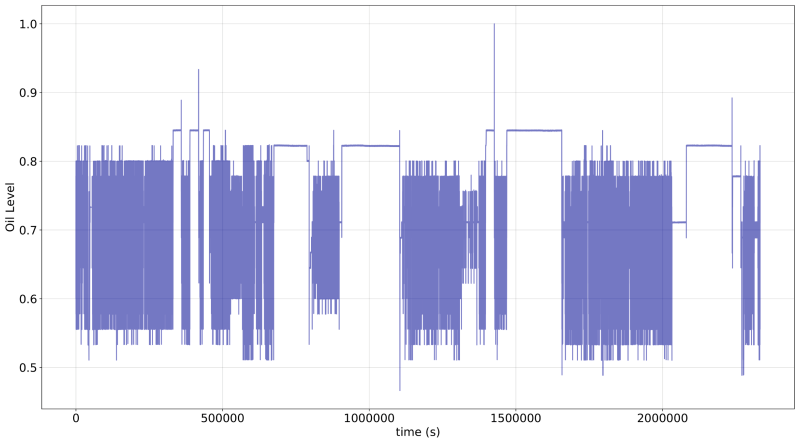

Fig. 1. Oil level.

1.1 Our Case Study

The case study is motivated by a typical maintenance activity in industrial plants, such as those of BENTELER Automotive. BENTELER produces and distributes safety-relevant products, serving customers in automotive technology, the energy sector and mechanical engineering. The production of such plants employs to a large extent machinery with several mechanical and hydraulic systems, which entail frequent and/or periodic maintenance such as lubrication oil replacement and refill, given that oil leakages are a common and expected phenomenon.

More specifically, our study focuses on early detection of oil leakage occurrences. Despite the fact that, typically, oil is mostly stored in large tanks equipped with oil level sensors, oil leakage detection is a challenging problem due to the continuous movement of oil across the machinery equipment parts. Such movement results in frequent increases and decreases of oil level, as depicted in Fig. 1. Therefore, and somehow counter-intuitively, simply monitoring the oil level is not adequate to provide concrete evidence about oil leakage.

To detect incidents, the main metric needs to be combined with two other types of information. Firstly, to process the oil level in relation to the concurrent IoT measurements of other aspects of the same equipment, such as temperature, current, mechanical part position and movement sensors, accelerometers, pressure sensors and so on. This allows oil level to be regarded in a specific context but has the drawback that not all other measurements are relevant. Secondly, to employ domain specific knowledge, which can assist the data analyst to understand whether the system is in idle state, as the oil level will be stabilized. The most straightforward way is to infer the time periods in which operation intermission takes place, when such information is not explicitly provided. For example, in such periods, some mechanical parts are often in a position in which they can never be during normal operation. Or, specific measurements of some

other parts are below some threshold. The challenge here is to automatically detect the data rules that define the operation status of the machinery before applying anomaly detection.

Overall, the goal is to provide a detection tool to the maintenance engineers, which can monitor and analyse at runtime key measurements of the system, in order to detect and report oil leakages at early stages. We consider that the measurement collection and fault reporting mechanisms are already provided, as the description of their development is out of the scope of this paper.

1.2 Summary and Paper Structure

In summary, in our case study, the aim is to detect oil leakage through combining oil level sensor measurements with other IoT device readings and inferring through data pre-processing the stabilized time periods. From the machine learning point of view, the goal is to devise unsupervised learning solutions that do not rely on a well-defined training set of past problematic states. The techniques are described in Sect. 2 and evaluated in Sect. 3. We conclude in Sect. 4. Non-essential technical details regarding our solutions are deferred to the Appendix.

2 Timely Failure Detection Approaches

2.1 Rule-Based Approach

Traditionally, thresholds are set to sensor measurements to trigger alerts on their violation. In the evaluation, a rule-based mechanism is used as a baseline approach. The simplicity of such a reactive approach is both an advantage and a disadvantage, as it is simple enough to be easily understood, developed and deployed, but rather naive to detect more complex events, which potentially encapsulate valuable information and to adapt in unstable environments, producing lots of false positive reports. As already explained in the use case presentation, due to the movement of the oil inside the machinery and the storage tank, the definition of a static rule for the detection of an oil leakage is a challenging task and if the requirement of an early detection is considered, more advanced solutions are required as explained hereby.

2.2 Outlier Detection Approach

Outlier detection is a vivid research field that has developed broad and multi-faceted algorithmic solutions. Through a variety of unsupervised learning solutions, ranging from basic clustering techniques like k-means [6], to Neural Network solutions like self-organised maps [8], we have selected an efficient and effective algorithm proposed in [9], namely the Micro-cluster Continuous Outlier Detection (MCOD) technique. For the challenging task of detecting outliers in data streams, MCOD provides low memory and processing footprint and its results are easily understandable, compared to the other solutions.

As the comparative study [12] suggests, the MCOD algorithm is considered as a state-of-the-art solution in the streaming data processing for distance-based outlier detection. Utilizing the MCOD algorithm in our case study, we can detect sudden changes (i) in the oil level or (ii) in other measurements related to the oil level, or (iii) combinations of measurements related to each other and to the oil level. The detected sudden changes may or may not be an indication of a fault.

MCOD on Raw Data. In this scenario, we apply the MCOD algorithm to the (normalized) raw measurements, without applying any filtering to the incoming data of a hot forming line. Maintenance experts have defined a short list of measurements (8 out of ∼1000) that might be correlated to the oil leakage use case. In the evaluation section we presents results from the application of the MCOD algorithm to a subset of these measurements. This setup increases the flexibility of the monitoring tool since it can be easily transferred to other use cases without the need of any prior knowledge of the production cycle.

MCOD Combined with Prior Domain Knowledge. Domain knowledge is important to interpret the machinery functionality and provide more specialized solutions, which can potentially give an edge over any other application agnostic proposal. In this scenario we have utilized domain expert knowledge to identify the status of the machinery (healthy/unhealthy) and obtain more stable results considering the oil level. Oil level is stabilized within 10 s after the machinery functionality is halted. Analyzing the incoming measurements considering the position of specific moving parts of the machinery, as in Fig. 2 (red line), we are able to define when the machinery is halted, and after a 10 s interval (green area of the Figure), apply the MCOD algorithm.

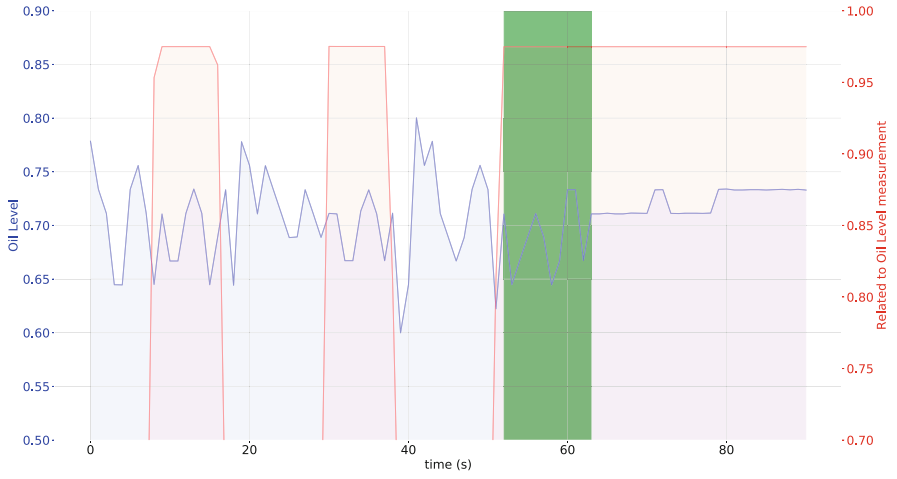

Fig. 2. Mechanical part movement in relation to oil level (Color figure online)

Algorithm 1. Outlier Reporting Filtering

```
minOutlierLife is a user defined parameter
po represents the last reported outlier
while true do
    sensor_meas ←fetch_measurements()
    outliers ←detect_outliers(sensor_meas)
    for all o in outliers do
        if o.life <minOutlierLife then
            if o.time >po.time + 5mins then
                report_outlier()
                po ← o
```

Practical Considerations. In distance-based outliers, a data point that has less than k neighbours inside a radius R, is an anomaly (see more details in the Appendix).

MCOD applies on data streams using a sliding window, hence there are user-defined parameters considering both the actual functionality of the algorithm and the streaming approach. There are four main parameters: (1) the window size W, which to either the time duration of the amount of the most recent data points considered; (2) the slide size S, which defines how fast/far the window moves in each algorithm step; (3) the threshold k on the number of neighbors in order a point to be labeled either as an inlier or an outlier in each step; and (4) the radius R that defines the radius of the neighborhood.

In continuous outlier detection, the detected outliers between all the active points are reported for every slide. If the slide size is lower than the window size (which is the usual case), then outliers will be reported multiple times. In a real case scenario of deployment of the algorithm in the production line, multiple reports of outliers can lead to frustration of the maintenance engineers. In addition, it is not practical to report all the outliers spotted in a short time range, as the maintenance engineers can physically investigate the machinery upon the first report of an outlier and usually the investigation requires more than some seconds or even minutes.

Hence, as it is presented in Algorithm 1, to provide a solution that is practically usable, we have defined a period of time (i.e. 5 min) after the first report of an outlier, where no other detected outlier is reported. We have also specified a parameter ($minOutlierLife$), which defines the amount of time (expressed in percentage of total number of slides inside a window) an outlier should be active before been reported.

3 Evaluation

The evaluation is based on real data obtained from the machinery of interest, working in the actual production line. The available data extend through 7 months. We have divided the available data into 2 datasets, which are examined separately. The first dataset, includes the first 4 months and its sampling

Table 1. Parameters for the first experiment

Parameter	Techniques			
	RL	MCOD	MCOD_DK	MCOD_M
meas.	oilLevel	oilLevel	oilLevel	oilLevel meas_1
W	-	3600	3600	3600
S	-	360	360	360
R	-	0.17	0.11	0.2
k	-	70	550	45
outlierLife	-	1	1	1
rule	<8800	-	-	-

rate is unstable but ground truth data exist. The ground truth reveals that in 13 days, maintenance tasks either explicitly or implicitly related to oil leakage have been reported. The second dataset includes the last month of the available data, where the measurements sampling rate is stabilized but there is no ground truth.

3.1 Proof of Concept Experimentation

In the initial set of experiments, we have used the first dataset and we have applied four variants: (i) rule-based (RL), (ii) MCOD on raw data (MCOD), (iii) MCOD with domain knowledge (MCOD_DK), and (iv) MCOD on multiple fields (MCOD_M).

The parametrization for each technique variant is presented in Table 1. The difference is in the R and k values. For the MCOD_DK approach, k is set to a higher value, while R to a lower than the other approaches, as the oil level measurement is much more stable, hence it is "safe" to set more strict thresholds to the algorithm (i.e. it will not produce too many outlier detection reports). Considering the R value, when more than one measurements are used for monitoring (i.e. the MCOD_M case), the R value should be increased, as the Euclidean distance is computed between pairs of points, not single points; otherwise, multiple false positives outlier reports will be created. For the RL approach the minimum acceptable value is 8300, however this threshold was never reached by the measurements of the date range of the experiment. Hence, we set the threshold to 8800, to maintain the number of the reported violations to an acceptable level, while detecting as much maintenance incidents as possible.

Table 2, presents the results for each one of the four experiments. In the Table, an X is placed if there is an outlier reported in the correct shift[1], a dash if there is no outlier reported and an $e[1|2]$, if an outlier is reported on an earlier shift (e1: one shift earlier, e2: two shifts earlier) than when it should be

[1] For D3 there are two maintenance tasks, hence we have used two places for an X divided by a slash.

Table 2. Results of the first experiment.

Dates	Techniques			
	RL	MCOD	MCOD_DK	MCOD_M
Day1	X	X	X	X
Day2	-	X	X	-
Day3	-	-/X	X/X	X/X
Day4	X	e1	e1	X
Day5	X	X	X	X
Day6	X	-	X	X
Day7	X	-	e1	X
Day8	X	-	e2	X
Day9	-	X	X	X
Day10	-	e1	X	e2
Day11	-	X	X	X
Day12	-	X	X	-
Day13	X	X	e1	-
FPD	3	15	10	12
#R	2790	56	32	39

reported based on the maintenance task logs. In the last two rows of Table 2, we present the number of false positive dates (FPD). FPD represents the number of days that the approaches reported at least one outlier unnecessarily. #R represents the total number of reported outliers. The lower the values of these two measurements, the better the possibility of acceptance of the approach by the maintenance engineers.

As it is observed, MCOD_DK achieves more balanced results, being able to detect all the incidents reported in the maintenance logs, with an exception of D4, D7, D8 and D12 where there was an outlier reported on an earlier shift. If this is considered a correct proactive warning, then the recall of the technique becomes optimal (i.e., 100%). MCOD also achieved remarkable results, if we take into consideration the unstable environment that it was applied on. Monitoring two measurements (i.e. MCOD_M) lowered both the number of FPDs and #R, detecting 5 more maintenance incidents and missing 2 compared to the MCOD approach. The RL approach produced the least number of FPDs, however it created too many violation reports (i.e. highest #R), missing the most maintenance incidents. The precision of MCOD_DK in terms of days remains above 50%.

3.2 Sensitivity Analysis

To demonstrate the sensitivity due to the R and k parameters, we present an experiment using the second dataset and the MCOD_DK technique. The used parametrization is presented in Table 3. The results are presented in Table 4,

Table 3. Sensitivity analysis parameters.

Parameters	Techniques		
	MCOD_DK1	MCOD_DK2	MCOD_DK3
meas.	oilLevel	oilLevel	oilLevel
W	3600	3600	3600
S	360	360	360
R	0.19	0.1	0.19
k	180	180	360
outlierLife	1	1	1

Table 4. Sensitivity analysis results.

Parameters	Techniques		
	MCOD_DK1	MCOD_DK2	MCOD_DK3
DA	9	19	14
#R	12	75	47

in terms of total number of days with alerts (DA) and total number of detected outliers (#R). The first parameterization is one achieved through combining fine-tuning and visual inspection of the raw data logs. This experiment reveals the most important weak point of an unsupervised learning approach, such as MCOD, namely its sensitivity to the input parameters: decreasing the R or increasing the k parameter (i.e. providing more strict parameters to the MCOD algorithm) creates more outlier reports.

4 Discussion

In this work, we considered a real industrial case study, where the main aim has been to detect oil leakages in a manner that (i) is automated; and (ii) can be used as an early notice for maintenance, in line with the PdM vision. In our techniques, we employed a state-of-the-art streaming outlier detection algorithm and we combined it with (i) domain knowledge and (ii) filtering mechanisms in order not to produce spurious or repeating results. The experimental results are particularly encouraging, given that we managed to report outliers for all or the immediate preceding shifts, where a maintenance task was reported.

The most important lesson learnt is that pure unsupervised learning techniques are inadequate to provide effective solutions. Domain knowledge, even in a simple form, can play a key role in devising techniques with very high accuracy and capability in detecting events in a timely manner. We have further verified this fact by repeating the experiments using another type of domain knowledge, where the machinery was in full operation and the results were similarly encouraging.

This work can be extended in several ways. For example, it can be combined with a machine learning-based predictor that can report on the estimated time of future failure. This direction calls for more holistic solutions based on an ensemble of techniques with complementary strengths. Also, further work is needed to render the technique less sensitive to its parameters capitalizing on existing work in the field of multi-parameter outlier detection, e.g., [4].

Acknowledgement. This research work is funded by the BOOST4.0 project funded by European Union's Horizon 2020 research and innovation program under grant agreement No 780732.

Appendix

Distance-Based Outlier Definition. A data point that has less than k neighbours inside a radius R, is called a distance-based outlier [7]. Figure 3 shows an example of a dataset that has two outliers with k = 4. The points *o1* and *o2* are outliers since they have 3 and 1 neighbours, respectively inside the R radius. In a data stream, we assume that we keep in a sliding window the most recent points, and the challenge is to continuously report all the outliers among the objects in that window. Apart from the technique in [9], additional streaming solutions are proposed in proposals, such as [3,5].

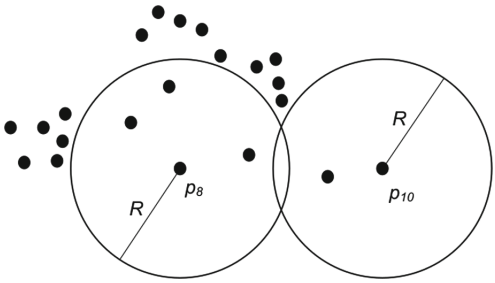

Fig. 3. Outliers example.

MCOD Algorithm. Finding the neighbours of each alive data object in a streaming scenario is a particularly computation-intensive process. The MCOD algorithm uses micro-clusters, as depicted in Fig. 4 (i.e. MC1. MC2, MC3) of radius $R/2$, inside which all the data points are inliers. Hence it alleviates the need of (re-)computing distances between all the data points on every window movement. The rationale is that, normally, most data points fall into one of such clusters and thus need not be further processed. Therefore, only a small portion of all objects needs to be examined. More details about the algorithm's functionality are provided in [9].

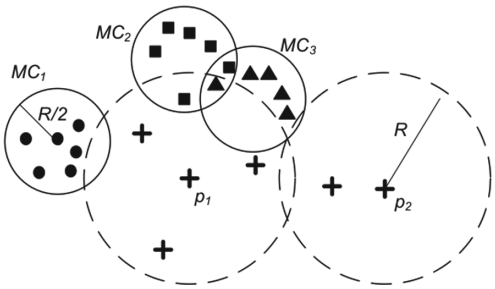

Fig. 4. MCOD - Micro-clusters.

References

1. Lu, Y.: Industry 4.0: a survey on technologies, applications and open research issues. J. Ind. Inf. Integr. **6**, 1–10 (2017)
2. Aggarwal, C.C.: Outlier Analysis. Springer, New York (2013). https://doi.org/10.1007/978-1-4614-6396-2
3. Angiulli, F., Fassetti, F.: Detecting distance-based outliers in streams of data. In: CIKM, pp. 811–820 (2007)
4. Cao, L., Wang, J., Rundensteiner, E.A.: Sharing-aware outlier analytics over high-volume data streams. In: Proceedings of SIGMOD, pp. 527–540 (2016)
5. Cao, L., Yang, D., Wang, Q., Yu, Y., Wang, J., Rundensteiner, E.A.: Scalable distance-based outlier detection over high-volume data streams. In: ICDE, pp. 76–87 (2014)
6. Hartigan, J.A., Wong, M.A.: Algorithm as 136: a k-means clustering algorithm. J. Roy. Stat. Soc. Ser. C (Appl. Stat.) **28**(1), 100–108 (1979)
7. Knorr, E., Ng, R., Tucakov, V.: Distance-based outliers: algorithms and applications. VLDB J. **8**(3–4), 237–253 (2000)
8. Kohonen, T.: Self-organizing Maps, vol. 30. Springer, Heidelberg (2012)
9. Kontaki, M., Gounaris, A., Papadopoulos, A.N., Tsichlas, K., Manolopoulos, Y.: Efficient and flexible algorithms for monitoring distance-based outliers over data streams. Inf. Syst. **55**, 37–53 (2016)
10. Korvesis, P., Besseau, S., Vazirgiannis, M.: Predictive maintenance in aviation: failure prediction from post flight reports. In: 2018 IEEE 34th International Conference on Data Engineering (ICDE), pp. 1414–1422. IEEE (2018)
11. Sipos, R., Fradkin, D., Moerchen, F., Wang, Z.: Log-based predictive maintenance. In: Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1867–1876. ACM (2014)
12. Tran, L., Fan, L., Shahabi, C.: Distance-based outlier detection in data streams. PVLDB **9**(12), 1089–1100 (2016)
13. Wang, J., Li, C., Han, S., Sarkar, S., Zhou, X.: Predictive maintenance based on event-log analysis: a case study. IBM J. Res. Dev. **61**(1), 11–121 (2017)