Anomaly Detection on Live Water Pressure Data Stream

Gal Petkovšek
Jožef Stefan Institute
Jamova 39, 1000 Ljubljana, Slovenia
gal.petkovsek@ijs.si

Matic Erznožnik
Jožef Stefan Institute
Jamova 39, 1000 Ljubljana, Slovenia
matic.erznoznik@ijs.si

Klemen Kenda
Jožef Stefan Institute
Jožef Stefan International Postgraduate School
Jamova 39, 1000 Ljubljana, Slovenia
klemen.kenda@ijs.si

ABSTRACT

We present the application of several anomaly detection algorithms to water pressure data streams. We evaluate their quality on unlabelled data sets using agreement rates. The applied algorithms are the Generative Adversarial Network (GAN), DBSCAN, Welford's algorithm and Facebook Prophet. We found that GAN performed best.

Keywords

water management, machine learning, anomaly detection

1. INTRODUCTION

In last decades, Internet of Things (IoT) has penetrated and shaped several fields such as energy management, traffic, health care and others. The water sector is, however, still implementing IoT solutions that will improve the water management with features such as real-time consumption prediction, leakage detection, water quality estimation and others.

In the presented work, we focus on the anomaly detection on the live water pressure data stream from the town of Braila (Romania). The overall goal of the research is to detect leakage points in the city's water distribution network. To detect the presence of a leakage in the system we apply an anomaly detection algorithm to the water pressure data stream. We considered several such algorithms, which were applied and evaluated on four data streams obtained from four pressure sensors. Our goal was to find the algorithm which returns the best results. Since the data is not labeled (regular or anomalous), the estimation of accuracy was done with a method considering relative agreement among selected algorithms [1]. The anomaly detection algorithms that were tested were GAN (generative adversarial networks) [6], DBSCAN [10], Welford's algorithm [9] and anomaly detection with Facebook Prophet [11]. It is important to note that first three algorithms consider the data stream as an actual live stream. This means that they consume one sample at a time (or a feature vector containing multiple past values, enrichment values and contextual data) and declare it regular or anomalous as the algorithms were intended to do in production. In contrast, the Facebook Prophet consumes the whole data stream as a batch and labels all the samples together. This makes it unusable in production (in this setting), however it is included in the experiment since it can help to estimate the accuracy of other algorithms.

Anomaly detection on time series is a well researched field. The algorithms in this paper were already considered in the related work in different settings and for different time series.

Anomaly detection can be used by estimating the expected regular interval in the upcoming measurement. This can be achieved in an incremental fashion with a simple short-term prediction model, for example with Kalman filter [7], or with a more advanced approach, based on time-series modeling [11]. The latter can be used in several settings, for example in detecting air temperature anomalies in the sewer systems [12].

DBSCAN [10] is a data clustering algorithm that can be applied in frequently changing data sets. Its incremental version [5] can be used in a streaming setting. The potential of the algorithm for anomaly detection has been demonstrated in several use cases, for example in detecting air temperature anomalies [3].

The paper that demonstrated the use of Generative Adversarial Networks for anomaly detection on data stream is fairly recent [6]. The authors have shown that this approach can outperform several other baselines on data sets obtained from NASA, Yahoo, Amazon etc. They introduced different measures of evaluating the reconstruction accuracy, which we tried to improve upon in our paper.

In this work, we use the already established anomaly detection approaches and compare their performance on an unlabeled water pressure data stream from a water distribution network. A more detailed description of the algorithms is given in the Methodology section. We argue that the relative agreement approach [1] improves the anomaly detection performance, which we demonstrate by manual evaluation of the results.

2. DATA AND DATA PREPROCESSING

We demonstrate our anomaly detection methodology on four data sets. Each of the data sets represents the pressure values of one of the sensors, which are located at different points in Braila's water distribution network. The sensors are labeled as '5770', '5771', '5772' and '5773'. The data sets contain between 10 and 11 thousand instances, which are spaced in 15 minute intervals, so about 100 days-worth of data. The data was first pre-processed to remove any duplicated points and 'holes' in the data which were formed as a consequence of sensor down-time. When working with data streams, this process should be done automatically to avoid any incorrect analysis when feeding the data into the anomaly detection algorithms. Each of the four data sets was split into a training and evaluation part. The training sets consisted of the first 2000 data points and the evaluation sets contained all the rest. This is done so that the algorithms which require training can be trained on one part of the data and evaluated on the other (GAN, DBSCAN).

3. METHODOLOGY

3.1 Evaluation of algorithms

Evaluation of the performance of algorithms on unlabelled data always represents a challenge. Since we are working with such data an actual calculation of accuracy scores would require manual labelling of the data instances. To avoid this time-consuming process, we use a method for estimating error rates (ratio of wrong classifications to the total number of instances) from the agreement rates of multiple algorithms. Agreement rate of two classifiers f_i and f_j is defined in the following way:

$$a_{\{i,j\}} = \frac{1}{S}\sum_{s=1}^{S}\mathbb{I}\{f_i(X_s) = f_j(X_s)\}$$

where $X_1, ..., X_S$ are unlabeled samples. The calculated agreement rates are then inserted into the following equations:

$$a\{i,j\} = 1 - e_{\{i\}} - e_{\{j\}} + 2e_{\{i,j\}}$$

Here we assume that the functions make independent errors we can substitute $e\{i,j\}$ with $e_{\{i\}}e_{\{j\}}$. With such a system of equations we can then calculate error rates using some root-finding algorithm. Such an approach has been previously used for the evaluation of classifiers on an unlabelled dataset [1]. Therefore we consider the anomaly detection algorithm as a binary classifier and use the aforementioned method for the comparison of different algorithms. Additionally, two important assumptions were made. Firstly, we assumed that the anomaly detection algorithms were independent and secondly, that each of those algorithms performs better than a random classifier.

Since the estimated performance of one algorithm depends on the output of the others it was important that the algorithms yield a similar percentage of anomalies. In other words, the algorithms are tuned to have similar predicted positive condition rate ($PPCR = \frac{FP+TP}{FP+TP+FN+TN}$). For most data streams this means that 1%-3% of the samples are labelled as anomalous.

3.2 GAN

The Generative Adversarial Network (GAN)[6] is an unsupervised machine learning approach to anomaly detection. An encoder-decoder structure of the neural network is used to first encode the input data point and then decode the encoded one. The model learns to reconstruct the input data point as closely as possible. The idea is that the reconstruction should be better if the input data is 'normal' and worse if it is abnormal/anomalous. We use an input vector, which is composed of 10 consecutive values of the uni-variate data stream. We then compare the input vector to the reconstructed one using the mean squared error (MSE) metric. We classify the data point as 'normal' if the value of the MSE is below the defined threshold. [6] calculated the thresholds using sliding windows on reconstruction errors (4 standard deviations from the mean of the window). We used a slightly different approach using the moving average multiplied by a constant as the threshold. This proved to be easier to implement on our live data stream use-case.

3.3 DBSCAN

DBSCAN [4] is a well-known data clustering algorithm. It groups together points, which are close together based on Euclidean distance. The group with the largest number of points in our case are considered 'normal', and the lower-density groups are outliers which are then labeled as an anomaly. The ϵ parameter which measures how close the points should be for them to still be considered of the same group, can be adjusted based on the data set, and the desired sensitivity of the algorithm. For DBSCAN we also use an input vector composed of consecutive pressure values. In this case, we discovered that a vector of 5-6 values works best.

3.4 Welford's algorithm

Welford's algorithm gets its name from the Welford's method for online estimation of mean and variance. A very simple anomaly detection approach [9] can then be constructed by defining the upper and lower limits (UL and LL) of ”normal” data as a function of mean and variance:

$$UL = mean + X * variance$$

$$LL = mean - X * variance$$

X is fixed and determines the threshold band. Any instance which falls out of that band is labeled as an anomaly. Instances can then be input into the algorithm one by one to be labeled and after each the mean and the variance (consequently UL and LL also) are updated.

For this experiment the actual Welford's method was not used since the mean and variance were computed from the last 1500 samples so that they would better adapt to the new samples. Note that the first 1500 samples therefore could not be labeled; however, this was not a problem since most of the other approaches required 2000 samples for fitting the models and the evaluation was therefore done on the remaining stream. However, the upper and lower limits of the interval were still computed as shown above with the value of $X = 2.2$.

3.5 Facebook Prophet

Facebook Prophet is an algorithm for time series forecasting that works especially well on data streams with multiple seasonalities [8]. Prophet also works well with missing data which makes it a good candidate for the problem at hand. After fitting the model it can make predictions for a chosen set of timestamps presented to it. Furthermore besides the prediction it also outputs upper and lower limits of the confidence interval for every sample. Ashrapov [2] demonstrates the implementation of an anomaly detection algorithm which uses this property to classify the samples inside the confidence interval as regular and the rest as anomalies. The model is fitted on the entire data set and then makes predictions on the same data set, providing both the anomaly detection and the confidence interval.

4. RESULTS

The results of the algorithms for data stream from sensor 5770 are presented in Figures 1, 2, 3 and 4. The charts show the raw values obtained from the pressure sensors, indicating the points which are labeled as anomalies with red points. Since the data sets are unlabelled it is hard to assess the accuracy of each algorithm based on anomaly visualizations alone, but we do notice some similarities and some differences. All of the algorithms are good at identifying obvious outliers (points which fall far out of the 'normal' range). The difference between the algorithms can be noticed when classifying points closer to the normal range. For example Welford's algorithm tends to label points as anomalies at the peaks of daily pressure fluctuation, which might not be ideal since we know that this behaviour can be considered normal. More sophisticated algorithms such as GAN and Prophet were also able to identify more "subtle" anomalies.

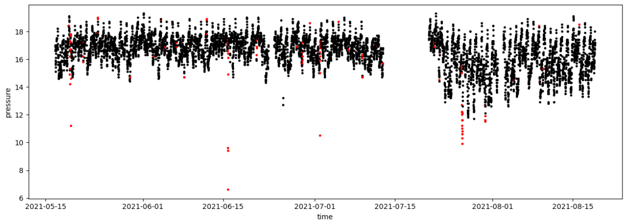

Figure 1: Anomalies found using GAN on data stream from sensor 5770.

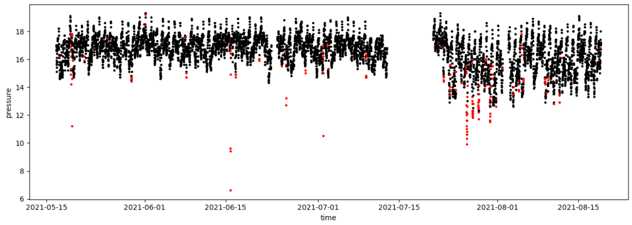

Figure 2: Anomalies found using DBSCAN on datastream from sensor 5770

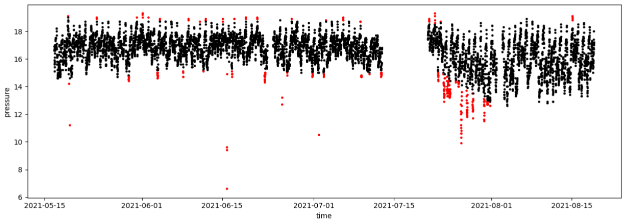

Figure 3: Anomalies found using Welford's algorithm on datastream from sensor 5770.

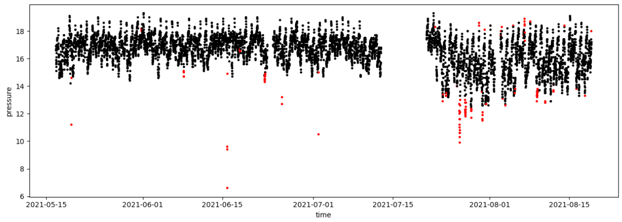

Figure 4: Anomalies found using Facebook Prophet on datastream from sensor 5770.

The recall of each algorithm can be increased or decreased by modifying parameters and thresholds. Since the data sets are unlabeled, it is hard to determine the optimal parameters. We decided to tune the algorithms to have similar recall of 1 - 3%, as we deemed that this would make the comparison of the algorithms the most fair. In Table 1 the shares of anomalies are presented for each separate data stream.

Algorithm	**5770 anomaly share**	**5771 anomaly share**	**5772 anomaly share**	**5773 anomaly share**
GAN	1.42%	0.99%	0.77%	1.13%
DBSCAN	2.63%	2.82%	2.73%	2.85%
Welford's algorithm	3.39%	3.41%	1.66%	3.16%
Facebook Prophet	1.66%	1.13%	0.46%	1.40%

Table 1: Shares of anomalies for all four data streams.

The error rates calculated from agreement rates are shown in Table 1 for each of the data streams. Since we assumed most of the samples in the data stream were normal these error rates are not very informative out of context. We can however, observe that Prophet performed best followed by GAN, DBSCAN and Welford, respectively. The results are consistent in all four scenarios. If we take into consideration that Prophet worked on the whole data set at once when the other three were limited to one sample at a time (as it is in production) we can declare that GAN performed best out of the algorithms that can detect anomalies on a live stream.

Algorithm	**5770 Error rate**	**5771 Error rate**	**5772 Error rate**	**5773 Error rate**
GAN	1.34%	1.38%	0.66%	1.09%
DBSCAN	1.59%	1.70%	1.78%	1.81%
Welford's algorithm	2.44%	2.41%	1.10%	2.31%
Facebook Prophet	1.14%	0.62%	0.39%	0.81%

Table 2: Error rates estimated from agreement rates for all four data streams.

We also considered a state-of-the-art method Isolation Forest, however it was too sensitive and therefore not usable in the error rate calculation.

5. CONCLUSIONS

We have tested five anomaly detection algorithms (Generative Adversarial Network, DBSCAN, Facebook Prophet, Welford's algorithm and Isolation Forest) on four separate data streams of water pressure data. Out of those five the Isolation Forest performed poorly since the share of anomalies found with this method was unreasonably high and was therefore not included in the final error estimates calculation.

Other approaches had similar shares of anomalies and were therefore used to calculate agreement rates and finally the estimated error rates of each anomaly detection algorithm. The results were consistent for all four data streams. Prophet performed best in every setting, however it looked at a data stream as a batch and it therefore could not be used for online anomaly detection. GAN performed second best followed by DBSCAN and Welford's algorithm which all work on a live data stream. Therefore we can conclude that the most fitting algorithm to be used for anomaly detection on the live water pressure data from water distribution network is GAN.

In future work, Facebook prophet could be adopted in such a way that it would also work on a live data stream since it has shown promising results in this experiment.

6. ACKNOWLEDGMENTS

This paper is supported by European Union's Horizon 2020 research and innovation programme under grant agreement No. 820985, project NAIADES (A holistic water ecosystem for digitisation of urban water sector).

7. REFERENCES

[1] Antonios Platanios, E. Estimating accuracy from unlabeled data.

[2] Ashrapov, I. Anomaly detection in time series with prophet library, Jun 2020.

[3] Celik, M., Dadaser-Celik, F., and Dokuz, A. S. Anomaly detection in temperature data using dbscan algorithm. *2011 International Symposium on INnovations in Intelligent SysTems and Applications* (2011).

[4] do Prado, K. S. How dbscan works and why should we use it?, Apr 2017.

[5] Ester, M., and Wittmann, R. Incremental generalization for mining in a data warehousing environment. In *International Conference on Extending Database Technology* (1998), Springer, pp. 135–149.

[6] Geiger, A., Cuesta-Infante, A., and Veeramachaneni, K. Adversarially learned anomaly detection for time series data, 2020.

[7] Kenda, K., and Mladenić, D. Autonomous sensor data cleaning in stream mining setting. *Business Systems Research: International journal of the Society for Advancing Innovation and Research in Economy 9*, 2 (2018), 69–79.

[8] Krieger, M. Time series analysis with facebook prophet: How it works and how to use it, Mar 2021.

[9] Lobo, J. L. Detecting real-time and unsupervised anomalies in streaming data: a starting point, Feb 2020.

[10] Schubert, E., Sander, J., Ester, M., Kriegel, H. P., and Xu, X. Dbscan revisited, revisited: why and how you should (still) use dbscan. *ACM Transactions on Database Systems (TODS) 42*, 3 (2017), 1–21.

[11] Taylor, S. J., and Letham, B. Forecasting at scale. *The American Statistician 72*, 1 (2018), 37–45.

[12] Thiyagarajan, K., Kodagoda, S., Ulapane, N., and Prasad, M. A temporal forecasting driven approach using facebook's prophet method for anomaly detection in sewer air temperature sensor system. In *2020 15th IEEE Conference on Industrial Electronics and Applications (ICIEA)* (2020), pp. 25–30.