doi:10.3233/FAIA210236

The Support System for Anomaly Detection with Application in Mainframe Management Process

Dominik STRZAŁKA[a], Alicja GERKA[a, 1], Bartosz KOWAL[a], Paweł KURAŚ[a], Grzegorz LEOPOLD[b], Michał LEWICZ[b] and Dawid JAWORSKI[c]
[a] *Department of Complex Systems, Rzeszów University of Technology, Al. Powstańców Warszawy 12, 35-959 Rzeszów, POLAND*
[b] *Z-RAYS, Plac Andersa 7, 61-894 Poznań, POLAND*
[c] *Department of Mathematical Modelling, Rzeszów University of Technology, Al. Powstańców Warszawy 12, 35-959 Rzeszów, POLAND*

Abstract. The process of mainframe machines managing and administration requires not only specialized expert knowledge based on many years of experience but also on appropriate tools provided by a machine performance management system, e.g. the Resource Measurement Facility (RMF). The aim of this paper is to show some preliminary results of Z-RAYS system construction that is built basing on machine learning (ML) techniques. It allows automatic detection of anomalies and generation of early warnings about some errors that can appear in the mainframe to support mainframe management process. Presented results are based on extensive simulations that were done basing on the IBM emulator. We focus on determining the degree of the metrics variability, the degree of the data repeatability in metrics, some approaches in metrics anomaly detection and solutions for event correlation detection in metrics.

Keywords. Mainframe, anomaly detection, support system, machine learning, Z-RAYS

1. Introduction

Mainframe (mainframe machines, big iron computers) are a class of computers used mainly by large organizations for critical applications like financial, statistical. Today, this term (name) is usually related to computers compatible with the IBM System/360 line introduced in 1965 [1]. There is no formal definition of a mainframe. Such machines run uninterrupted for very long periods of time and are used everywhere when the high availability is required because any downtime would be costly or even catastrophic. They can run in parallel different instances of operating systems thanks to the technique of virtual machines. They are not a supercomputer [1].

In this paper we focus on a series of preliminary studies and analyses to prepare:

[1] Corresponding Author, Alicja Gerka, Department of Complex System, Rzeszów University of Technology, Al. Powstańców Warszawy 12, 35-959 Rzeszów, Poland; E-mail: a.gerka@prz.edu.pl.

- recommendations for metrics,
- proposal of suggested algorithms and possible approaches to anomaly detection,
- preparation and presentation of the results of the performed series of experiments,
- preliminary studies on correlating events that are anomalies.

The aim of this paper is to solve three main problems and show some preliminary solution results. The first problem to be solved is the development of a methodology for detecting potential anomalies in time series representing the metrics. One possible approach to solve this issue is to only refer to statistical analyzes, but another approach is to rely on ML algorithms related to time-series analyses. Bearing in mind the fact that from a mainframe machine more than 15 thousand RMF metrics (but not only) can be constantly analyzed, an approach to automatize this process is expected, in particular making decisions on how the data from the metric will be processed. The second aspect is to identify one or more approaches that allow the use of different anomaly detection methods together with preliminary research of these solutions. This approach is about the proper detection of anomalies, also with the indication of the moment when such an anomaly value appears or there is a significant deviation from the adopted metric pattern behavior. The third element is related to the choose of solution for correlating events (anomalies in time series) and the detection of hidden dependencies (or correlations) between these anomalies.

The research and development problem is: how, for collected, visualized, and analyzed data, to create a support system that allows to take the right administration decisions and ensure uninterrupted mainframe work.

2. Preliminaries and methods

The source of all provided metrics is a special mainframe emulator (IBM Z Development and Test Environment) and RMF, a mainframe performance management tool that measures selected areas of system activity and presents the collected data in the form of metric records [2].

In our carried study, the supplied dataset contained 42 metric datasets generated on a mainframe emulator with the following unique SOURCEID's like 8D30F0, 8D0360, 8D1040 which showed selected grouped metrics. Each such metric can consist of many time series and as part of each set of metrics (SOURCEID), data on various types of devices (DEV) and processes (RCNAME) are presented via time series with different time resolutions. From the first raw data, it was possible to generate over 500 unique metrics in the form of time series containing numerical data with a time stamp.

The provided metrics under the name RPNAME (RePort Name) are very extensive, containing many columns of various data, but from our point of view, the most important columns are VALUE and dateAndTime. Other relevant columns may indicate: DEV - external device name, SYSTEM - system name, RCNAME - MASTER, OMVS and their correct interpretation leads to the final metric suitable for analysis in the form of a time series with VALUE and dateAndTime.

We decided to take the resolution of 1 minute for each stored data. Originally, the data was saved as a text file, but for large systems, it could mean the need to collect up to more than 500 GB of historical data per year. Using a different recording format, e.g.

binary, assuming a regular (every minute) recording only the metric values in a bit format (4 bytes per number) were stored, and in the absence of a byte record the value –1 was signed. The total amount of data was significantly limited to 5,760 bytes per day per metric. It should be noted here that many RPNAME metrics (Table 1) contain many data relating to different DEVs within the same timestamp (so-called splitting).

Table 1. Some of the original set of RPNAME metrics for the studies

#	SOURCEID	RPNAME
1	8D16D0	delayforXCFbyWLMreportclass
2	8D0E90	Ioactivityrate
3	8D12A0	Iointensitybyvolume
4	8D12C0	IOSqueuetimebyvolume
5	8D27B0	MSUbyWLMreportclass

2.1. Determining the degree of the metrics variability

After the first month of data gathering, graphical visualizations of metrics were done. They showed that some of them are constant (do not change during the time) or contain only two values, -1 and 0. Mainframe experts decided that this could be a normal behavior related for instance to the fact, that some processes in the mainframe machine can be visible only for short periods, several times per year, or when the system is in a normal state. We decided to classify all metrics to select those, which have high volatility. Among the numerical characteristics of doing such classification, it should be mentioned for example: analysis of the histogram, searching for a modal value, and testing the power spectrum to determine periods (days, weeks) of repeating certain processes in the metrics. This problem is critical to the proposed ML algorithms.

One of the proposed ways to determine the degree of variability (1) of the metric may be to examine the process increments:

$$D_V(t) = Y(t+1) - Y(t), \qquad (1)$$

where: $Y(t)$ is the value of metrics in t – time, $Y(t+1)$ is the next value of metrics in t – time.

In this way, each increase/decrease in the value of the metric will be visible in the new time series and the periods when the metric has constant values (not necessarily = 0) will be treated as no variation. Then, for a new time series of process increments, the modal value can be determined. Keeping in mind that the system was observed for 7 days, 24 hours a day with a resolution of 1 minute, it generates 60 * 24 * 7 = 10,080 data per metric (time series) during one week. For example, if for more than 99.9% of cases the modal value will be 0, then the time series has a very low variation, 99.9% of cases equal to 0 give ~10 changes in the value of the metric during the week and ~1.5 per day. It was called Category A. Details are in Table 2. This approach can be changed: more categories can be added or different values for reference points determined.

Table 2. Proposed degree of the metrics variability

Modal value x for 0	Category	Changes of values in metrics
More than 99.9%	A	< 1.5 per day
$99.9\% \leq x < 99\%$	B	< 15 per day
$99\% \leq x$ 90%	C	< 150 per day
Other	D	more than 150

2.2. Determining the degree of the data repeatability in metrics

The visual analysis for selected metrics indicates a certain cyclicality of the processes; determining the period of this cyclicality may be of key importance for selecting the parameters of ML algorithms. One way to detect and quantify this cyclicality is the (power) spectrum of the metrics. The occurrence of distinct power peaks for specific frequencies in the power spectrum will suggest that some characteristic processes of a specific period (inverse of the frequency) are occurring. This period can be expressed in hours, days, weeks. The fast Fourier transform was used [3] and some results of this approach are visible in Figs. 1 and 2, where the frequency axis (Fig. 2) is given on a log scale (the lowest values of frequency are the most important because they indicate the longest repetition time; on a log scale they are better visible).

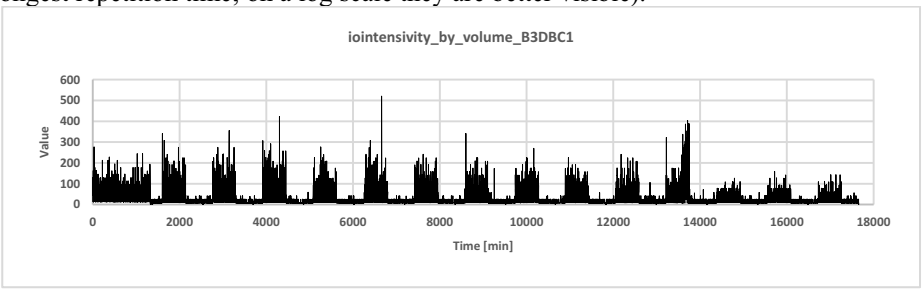

Figure 1. Time plot for the iointensitybyvolume-B3DBC1 metric for 15 days.

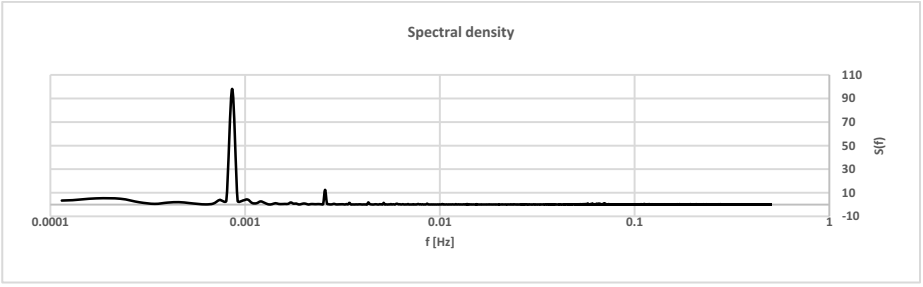

Figure 2. Spectrum $S(f)$ for the iointensitybyvolume-B3DBC1 metric with one visible dominant frequency.

In the used method one can calculate that the number of process repetitions is equal to the number of collected observations multiply by the dominant frequency f, where
$f \in < 0, 0.5 >$. Having in mind that during one day there are 1440 minutes we can see that: the number of collected data/1440/the number of repetitions determines how many days the process is repeated. This defines the TIME_STEPS parameter in ML algorithm. In Fig. 2 there is one distinct peak for f = 0.000857339 Hz. Given that 17498 input data of the time series were analyzed, it can be calculated that 17498 * 0.000857339 = 15. This means that there are 15 repeating process periods in the input series, and the series shows 15-day data, i.e., a specific process is repeated once per day.

3. Metrics anomaly detection

In our approach we proposed to use ML based algorithm that firstly learns how analyzed metrics behave in normal state and then is able to find anomaly behavior of time series. We tested 3 different algorithms with different parameters [4][5][6].

Table 3. Summary of tested variations of algorithms.

	Parameter				
Algorithm	**GPU (Computing power)**	**TIME STEPS**	**Loss (MAE vs MSE)**	**Dropout**	**Change in the number of neurons**
Autoencoder with 2 LSTM layers of 128 neurons each	Average run time for metrics is 37 s.	Min 50	Use MAE	Suggested 0.2-0.3	Not tested
Single layer LSTM network (128 neurons)	Runs 2x faster than alg. 1	Min 50	Use MAE	Not tested, unable to test	Min 32 neurons
Convolutional autoencoder	Runs 2x faster than alg. 1	Min 50	Use MAE	Suggested 0.2-0.3	Not tested

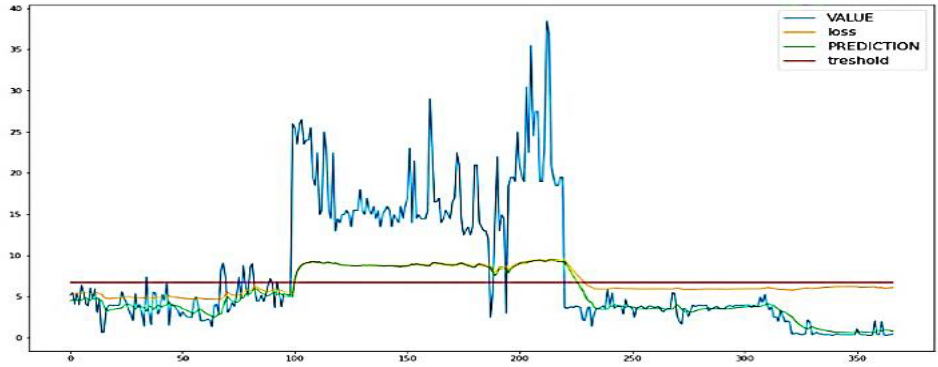

Figure 3. Blue line – value of the metric, the orange one – the loss function, the green one is a prediction, the red one is a threshold.

The purpose of the algorithm is to load data from metrics in order to detect anomalies. In the first cycle of the algorithm, the raw data of the metrics are loaded and divided in the ratio of 80/20 (training data and testing data). After the standardization and transformation of the data by the scaler, the function No. 1 (Autoencoder with 2 LSTM layers of 128 neurons each, with dropout 0.2 and TIME_STEPS = 50) was used. This algorithm generates the most likely results after the consultation with the mainframe server administrators. Basing on these results the following approach was proposed: 1) calculate basing on the analysis of repeatability the value of TIME_STEPS parameter, 2) basing on chosen ML algorithm data calculate the loss function as a difference between real data and predicted, and use the threshold as a max value of loss function. If the loss value is above the threshold generate the anomaly. The final results of this approach are seen in Fig. 3.

4. Proposed solutions for event correlation detection in metrics

The so far proposed approach allowed for detection of anomalies in one metric, but there are many metrics and between them some correlations can exist. We can find them with several approaches. The simplest one is based on Pearson's linear correlation coefficient *r* [7], but the number of comparisons will be approximately *n(n-1)/2*, so for 700 time series, this is about 250,000 comparisons. This solution can produce a matrix that is symmetric about the main diagonal (values of 1 on the diagonal), which will indicate in the range -1 ... +1 the degree of correlation. A preliminary analysis took more than 16 hours to compute the matrix (available at [8]). The second approach can be based on the idea of supermetric - the metric of all metrics.

4.1. 'Supermetric' - the metric of all metrics

As part of this approach, a solution is proposed to do a metric that synchronous records the current state of the system. If an anomaly occurred in several metrics (m_1, m_2, … m_n) the value of supermetric in the form of a mathematical formula will be calculated (see Fig. 4).

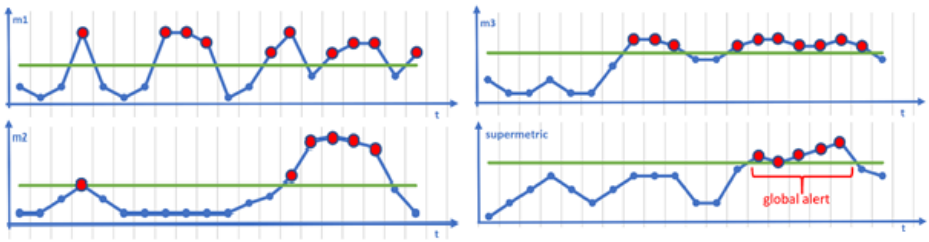

Figure 4. Overview drawing of the 'supermetrics'. Green line is a threshold, red dots are anomalies.

Such a supermetric can be also analyzed via ML algorithm also to determine the trend of anomaly behavior (toward deepening anomalies, increasing their number, etc.). However, under this approach it is hard to find the exact mathematical formula for all existing metrics. In order to solve this problem we decided to use the following approach: having in mind that some metrics exhibit similar behavior we take the cluster analysis to find the clusters of metrices with similar statistical parameters (Fig. 5). According to [9] the key point is to find the right measures of distance that influences the final results. The carried experiments showed that the Ward method is the best one [10, 11].

Basing on these results we build for each cluster a new $A_K(t)$ function (2) that for each cluster measures the possibility of second row anomaly in clusters:

$$A_K(t) = \frac{\sum_{n=1}^{n} \tanh(F_n)}{n}, \qquad (2)$$

where: F_n – is the function of anomaly detection for each metric in the cluster, n – the number of metrics in the cluster, t – time.

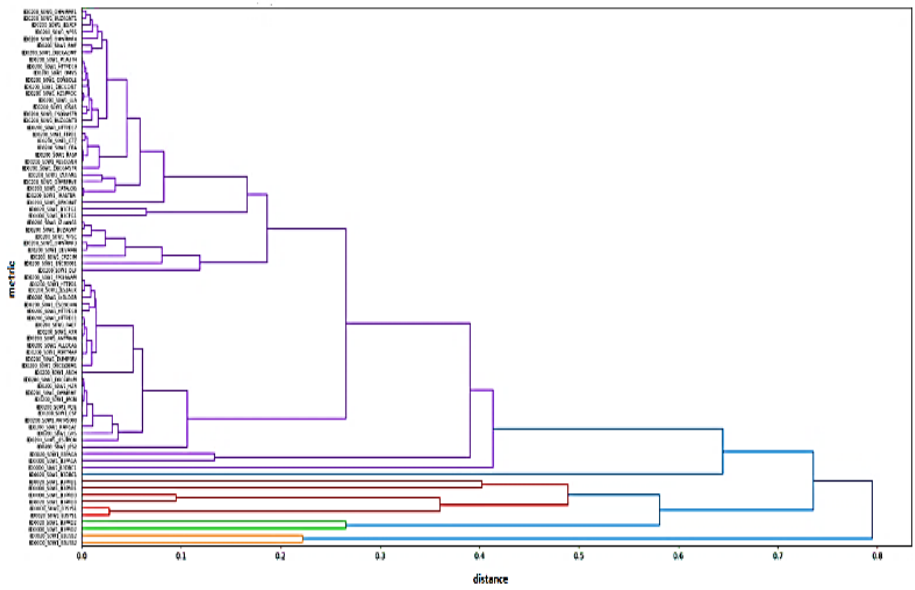

Figure 5. The results of metrics clusterization.

For each cluster with defined A_K function we use the isolation forest algorithm to determine possible anomalies. The results are seen in Fig. 6.

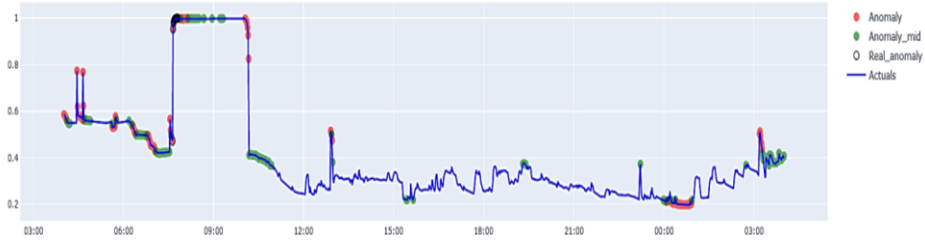

Figure 6. Detection of second row anomalies in defined clusters.

5. Conclusions

In this paper we show some preliminary results of R&D activities related to the Z-RAYS support system for anomaly detection with application in mainframe management process. There is a lot of work to be done in the future comparing to the other similar solutions like Elastic Search [12] in order to guarantee that our proposed system will have the high specificity and sensitivity. The so far obtained results convince that proposed solutions can efficiently handle with the problem of automatic anomalies detection. The main research and development limitations are related to the computational power available for the whole support system. It is obvious that one can build a very sophisticated system that will require a huge amount of resources, however the results given by such a system will be no better that those guaranteed by a human operator.

6. Acknowledgements

This paper was partially supported by Polish National Centre for Research and Development (NCBiR) under agreement 9/2020, R&D project Z-RAYS – MAINFRAME SYSTEMS MONITOR.

References

[1] https://www.ibm.com/docs/en/zos-basic-skills?topic=concepts-value-mainframe-today
[2] https://www.ibm.com/products/z-development-test-environment
[3] H. Musbah, M. El-Hawary, H. Aly, Identifying Seasonality in Time Series by Applying Fast Fourier Transform, 2019 IEEE Electrical Power and Energy Conference (EPEC), 2019, pp. 1-4.
[4] G. Hackeling. Mastering Machine Learning With scikit-learn. Packt Publishing. (2014).
[5] F. T Liu, K. Ting, Z-H. Zhou, Isolation Forest. ICDM '08. Eighth IEEE International Conference on Data Mining, pp.413 - 422.(2009).
[6] M. Togbe, et al. Anomaly Detection for Data Streams Based on Isolation Forest using Scikit-multiflow. The 20th Int. Conf. on Comp. Sci. and its Appl. (ICCSA 2020), Caligari, Italy.
[7] J. Freeman and T. Young. Correlation coefficient: the association between two continuous variables. https://www.sheffield.ac.uk/polopoly_fs/1.43991!/file/Tutorial-14-correlation.pdf.
[8] https://github.com/pawkuras/SSFAD/blob/main/Table1.docx
[9] S. Saraçli, N. Doğan, I. Doğan, Comparison of hierarchical cluster analysis methods by cophenetic correlation. J. Inequal. Appl., 203 (2013).
[10] S. Hands, B. Everitt. A Monte Carlo Study of the Recovery of Cluster Structure in Binary Data by Hierarchical Clustering Techniques. Multivariate Behav Res.22(2):235-43, (1987).
[11] Murtagh, F., Legendre, P. Ward's Hierarchical Agglomerative Clustering Method: Which Algorithms Implement Ward's Criterion?. J Classif 31, 274–295 (2014).
[12] D. Turnbull, J. Berryman. Relevant search: with applications for Solr and Elasticsearch, Manning Publications Co., (2016).