Applying Machine Learning to Study Infrastructure Anomalies in a Mid-size Data Center – Preliminary Considerations

Piotr Janus, Maria Ganzha
Mathematics and Information Sciences
Warsaw University of Technology
Warsaw, Poland
m.ganzha@mini.pw.edu.pl

Artur Bicki
EMCA Software Sp. z o.o.
Warsaw, Poland
artur.bicki@it.emca.pl

Marcin Paprzycki
Systems Research Institute
Polish Academy of Sciences
Warsaw, Poland
marcin.paprzycki@ibspan.waw.pl

Abstract

Today, data centers deal with fast-growing data volumes. To deliver services, they deploy a growing amount of heterogeneous hardware. As a result, it becomes practically impossible to apply human-based data center management. For instance, in a real-world data center with 500+ computers – delivering data, computational, and network services, it becomes impossible to visualize and understand causal relationships among variables describing the performance of monitored resources. However, it is possible to collect data describing the behavior of individual nodes. Hence, such data may be used to analyze/model system performance. In particular, it may be applied to recognize and predict anomalies in system behavior. Furthermore, collected data should allow finding the cause(s) of anomalies. Therefore, "data-driven approaches" have been applied to the real-world data, to find Root Cause of anomalies.

1. Introduction

One of the key developments of the "second wave of the Internet" are ecosystems of connected resources (sensors, actuators, gateways, data repositories, servers, etc.). Their nascent results in the growth of the volume of generated, stored, and processed data. As stated in [1], analysis of Big Data can bring business to the next level, allowing better decision-making, cost optimization, and other benefits. However, access to available data has to go in tandem with the delivery of new analytical tools.

The prevalent approach of dealing with data deluge is based on cloud computing. As a result, big cloud providers (e.g. Amazon, Google, and Microsoft) started to dominate the landscape. Nevertheless, numerous companies, SMEs in particular, facilitate data managing infrastructures. They work, mainly, with "local businesses". While there exist several reasons why companies elect use of local data centers, addressing them is out of the scope of this work.

EMCA Software is a vendor of the Energy Logserver (EL) system, which collects data concerning IT infrastructure metrics. In actual customer deployments, EL collects metrics from hundreds, up to thousands of devices. Working in many sectors, e.g. finance, insurance, energy, retail, EL collects data from streams, with a speed of a hundred thousand values per second. Note that, the trend of enterprise IT is to centralize data, and collected metrics/logs, in one place. Specifically, values related to CPU utilization, disk space occupancy, and application performance, among others, are brought together, to observe trends and baselines. EMCA, as a solution provider, has as its goal to help enterprises to detect suspicious changes in performance data, especially when they can influence business continuity and its expected level.

The underlying assumption behind the presented work is that it may be possible to use performance data to predict anomalies and their causes. Specifically, *Root Cause Analysis* (RCA) concerns the situation when an anomaly occurred within the data center. The goal of the data-driven investigation is to establish its actual cause. Moreover, after establishing what was the source of a given problem, a way to prevent future faults from occurring can be sought. However, the latter is out of the scope of the current contribution, which presents the initial results of an ongoing investigation.

In this context, a discussion of data that is collected by EMCA can be found in Section 2. Once characteristics of available data are presented, the related works are discussed (Section 3). Systematic discussion on undertaken approaches follows in Section 4. Finally, in Section 5, a brief explanation of obtained results, and an outline of the future research directions, are provided.

The main contributions of this work are: (1) The proposed solution to anomaly detection in data center streaming data outperforms current state-of-the-art models. (2) A scalable end-to-end solution that works in unsupervised environment, to perform RCA is proposed.

URI: https://hdl.handle.net/10125/70636
978-0-9981331-4-0

Moreover, the presented RCA process is unique for Big Data ecosystems, such as data centers.

2. Available data as the "driver"

Let us start by discussing data collected by EMCA, in its installed systems. Here, three types of resources can be identified: computational, data, and network servers. However, this distinction is not considered. This is related, in large part, to the fact that all servers work as, so-called, "analytical groups" serving customers, applications, etc. Therefore, an analytical group (of servers) is treated jointly (see, Section 4.1). Table 1 contains an overview of the dataset used in this study. Note that this Table summarizes data from 2 weeks. Therefore, the size of one full year of data would be $\sim$ 13 TB. This number should be kept in mind when considering methods, that could be applied. While it does not preclude any anomaly detection method, but it is a realistic "constraint" that has to be dealt with.

Volume	$\sim$ 500 GB
Time period	14 days
Time-step	5 minutes
Number of monitored resources	537
Number of monitored services	$\sim$ 122 thousand
Number of data instances	$\sim$ 495 milion

Table 1. Meta-level summary of the complete EMCA Dataset.

In machine learning, three situations can be distinguished (see, [2], Chapter 15). (1) When the training data is tagged; capturing **all** situations where behavior(s) are to be distinguished. Here, *supervised learning* can be applied. (2) When some behaviors are tagged, while others are "not known to be present". A typical example is when a distributed denial of service (DDOS) traces are tagged, and used for learning, while the system should generalize, to recognize other DDOS cases. This brings *semi-supervised* learning. (3) Finally, *unsupervised learning* is applied to "raw data" (no tags available). This applies to the EMCA Dataset.

Currently, EMCA anomaly detection is based on an assumption that an *occurring anomaly results in unusually high readings of selected parameters* (exceeding established thresholds). However, as shown in, for instance [3]), an unusually high reading of a parameter may not mean that an anomaly occurred. Thus, such an approach is inadequate.

Note that one should distinguish between: *point*, *contextual*, and *collective* anomalies. *Point anomaly* occurs when a data instance significantly differs from others, within the same time series. Such instance can be compared either to the other values in the time series (globally) or to its closest (in time) data instances (locally). Here, an example is a sudden, unexpected, increase of node usage, observed for a short time. *Contextual anomaly* occurs when data instances can only be considered abnormal when viewed against some meta information. Here, temporal or spatial information, for a given time series, can be used. An example of an anomaly, with a temporal context, is when during peak-hours, expected high usage of CPU or data transfer is not found in the data. Finally, *collective anomalies* are recognized when to define whether an anomaly occurs or not, it has to be confronted with other reference time series. Individual data instances may not be anomalous, but their co-occurrence with others indicates that the "group is anomalous". An example concerns a group of cooperating nodes. Here, higher (varying) use of some nodes is considered to be typical behavior. However, when all computing units reach maximum load, then a collective anomaly occurs.

Therefore, static "critical threshold(s)" related to one, or more, individual parameter(s), result in detecting *point anomalies*, while leaving other anomaly types untreated. Moreover, exceeding a single threshold (or a few of them) may not indicate an anomaly. Hence, the current approach may lead to incorrect indicators (false-positives, detecting a non-existing anomaly, or false-negatives, not detecting an existing anomaly). To deal with the incorrect anomaly detection, and to correctly recognize (only) cases when there was a problem, expert knowledge would have to be applied. However, this is not an easy task, even for the best specialists.

Recall that the collected data may originate from multiple resource types. Moreover, nodes belonging to a single "resource category", are also heterogeneous. This is because they have been purchased (added to the infrastructure), over a certain period (e.g. 3+ years). Hence, what for one node will be an "alarming behavior" (e.g. running CPU utilization close to 100% on the newest server), may be quite normal on other machines (the oldest nodes, in the infrastructure).

As noted in Table 1, data is collected "every 5-minutes". Hence, 35 Gbytes of data, generated each day, need to be analyzed. This data originates from 537 nodes, which can influence each-other. Here, even if it would be possible to draw a graph of interactions among 537 nodes, understanding its meaning becomes extremely difficult. Moreover, each node is likely to exhibit its periodic behavior. Thus, it might be extremely hard, if not impossible, for an expert to notice a "strange behavior" with so many others (co-)occurring in the system; and tag it as anomalous.

Finally, while it is easy to "capture" a major failure,

catastrophic events occur rarely. Hence, they do not deliver enough training data, even for semi-supervised machine learning. No company should wait for a major crash, to look for the cause of problems.

Overall, experts may not uncover all anomalies (they may hide in deluge of data, in complex behaviors of individual nodes, and/or relations between nodes in the ecosystem). Moreover, tagging data, even for semi-supervised approaches, would require enormous effort (and costs), while its performance would be limited by human capabilities (to analyze complex problems, often under time pressure). Therefore, the raw data RCA will have to be performed and unsupervised machine learning applied.

It should be noted that, as mentioned in Section 3, in similar cases, i.e. finding anomalies in data centers/clouds, researchers reported their results based on the "Yahoo! Dataset" [4]. This dataset consists of four subsets (A1-A4), containing real and synthetic time series. Moreover, each data point is tagged as anomalous (or not). Yahoo! Dataset allows for anomaly detection performance benchmarking. In the synthetic subsets, multiple scenarios are represented, e.g.: non-stationarity (by a change in mean value, or variance); high noise; and periodicity. Nevertheless, the most important is the A1 series, representing metrics from the actual Yahoo! computational services. However, it is not directly specified what specific metrics are present in this dataset. A detailed summary of the datasets can be found in Table 2. Supervised learning can be applied to it. This fact will be used in this work (see, Section 4.3).

Dataset	No. time series	Characteristics
A1	67	Real world data from Yahoo! cloud infrastructure
A2	100	Synthetic data with seasonal, noise, trend components. Contain mostly point anomalies.
A3	100	Synthetic data with characteristics as A2 datset with higher noise contribution.
A4	100	Synthetic data with A2 components changing over time.

Table 2. Meta-level summary of the Yahoo Dataset.

3. Related work (SoTA)

First, recall that only "raw data" is available and thus, unsupervised learning has to be applied. This restricts the scope of the SoTA considerations. Moreover, only algorithms developed for Big Data analysis are presented. Finally, the key aspect of research is to investigate whether the application of standard RCA approaches is possible to data center size systems.

In "smaller systems", where the number of devices and their properties is limited, *data reconstruction* has been successfully applied. The idea is to encode the original data in a smaller latent space, and reconstruct it "back". A high reconstruction error may indicate an anomaly. Here, one of the popular models is the Principal Component Analysis (PCA). For instance, a solution found in [5], employed PCA to detect anomalies in manufacturing. However, it's hard to use the PCA in a data center, because of dynamic changes and nonlinear relations within data. Moreover, establishing the "right number of components" (for large real-world data) is a research topic on its own (see, Section 4.2).

A remedy might be an Autoencoder (AE [2]) neural network architecture. AE models consist of: an *encoder* that maps the input into the latent space, and a *decoder*, which reconstructs the input data. Autoencoders should catch nonlinear relations between variables. A comparison between PCA and AE models presented in [6], indicated the primacy of Autoencoders. Similar approach has been described in [7]. Variational Autoencoders [2] were used to keep track of anomalies occurring in network traffic, in data center systems.

Despite advantages, reconstruction models have significant shortcomings. (1) In real-world systems, data is collected with "different timestamps" (individually, by each device). Hence, the reconstruction models have to "wait" for *complete set of observations*. (2) Thus, it might be difficult to apply them to stream data analysis. This is crucial when the system is "dynamic" and its state changes very fast. If anomalies have to be detected "fast", then reconstruction models may not work. (3) Moreover, with the growth of the system, the reconstruction time is also growing. For instance, the complexity of the PCA is $O(N^4)$ (where N is the size of the input data), whereas training neural networks is known to be unpredictable, and very time-consuming.

As a result, methods for soft real-time anomaly detection were proposed. They focus on individual time series. One of the simplest models uses an iterative implementation of the classical ARIMA [8]. Since an autoregressive model is a special case of ARIMA, the iterative estimation of model parameters can be applied. Performance of the Online ARIMA was tested on data from a cloud environment, with seven computational, and three controller nodes. The proposed model achieved a low false-positive rate. A known drawback of this solution is its difficulty in handling of seasonal time series, with variation changing over time.

Approach drawn from image analysis was employed in [9], and used in cloud infrastructure

anomaly detection. Here, Fourier transform is used for calculating Spectral Residuals, a compressed representation of time series that emphasizes its unique parts. Next, an inverse Fourier transform is applied, delivering so-called, *saliency map*. It provides an anomaly score for each time step. By comparing with a static threshold, each data instance can be classified as "normal" or "anomalous". Here, the performance of the presented model was tested on two datasets: the Yahoo! Dataset, and an internal private dataset with similar characteristics. However, while this method does well with stable and seasonal time series, it underperforms for noisy data. Moreover, for longer (persisting) contextual anomalies, it tends to indicate only the beginning of an anomaly. This might be misleading, as one might see it as a point anomaly, while the actual problem is more complex.

Another approach is based on the Extreme Value Theory (EVT, [10]), and was used in network traffic analysis (see, [11]). Here, anomaly detection is based on modeling the tail of the data distribution. Two models of anomaly detection are proposed: (i) SPOT, for stationary time series, and (ii) DSPOT, capturing concept drift. For streaming data, both models employ Peaks-Over-Thereshold [12] method, for tail distribution estimation. Overall, EVT-based methods work best with point anomalies. They underperform for seasonal time series and contextual anomalies.

For seasonal and periodic time series, the DeepAnt approach, based on convolutional neural networks (CNN, [2]), was proposed ([13]). In every iteration, DeepAnt predicts the next timestep. Based on the Euclidean distance, between real and predicted values, data is considered anomalous, when it exceeds a chosen static threshold. A shortcoming of this approach is the complexity of the model, preventing immediate parameter adjustment, in case of sudden concept drift.

Autoencoders can be used not only to reconstruct the whole system but can also be applied to individual time series. As shown in [14, 15], VAE can be applied to monitor web application services. Introduced models try to capture "normal characteristics" of the time series, and to recreate them. Different configurations of models can be developed, taking advantage of deep learning (e.g. the LSTM [2]). The key advantage of this approach is its robustness. Specifically, even when anomalies are present in the training dataset, the model can capture the "normal behavior".

Finally, a solution was introduced by LinkedIn, to monitor its cloud infrastructure. Based on [16], Luminol [17] software was developed. Proposed model segments the time series into chunks. Then, using the frequency of similar chunks, the anomaly score is calculated. This model is assumption-free, however, it is a seasonal time series oriented.

Overall, a number of models and approaches can be used in anomaly detection, in data center monitoring. Nevertheless, in the case of unsupervised learning, there is no "ultimate solution", that meets all requirements of data centers. Therefore, we have proceeded with our investigation, seeking an anomaly detection approach that can be applied in EMCA practice.

4. Experimental results and their analysis

Thus far it has been established that this study deals with a relatively large dataset, with no tags for anomalies. Furthermore, key analytic techniques, that can be applied to the problem, have been emphasized. Now, let's discuss the results of the conducted experiments.

4.1. Experimental setup

In experiments, both the Yahoo! Dataset and the EMCA Dataset were used. The Yahoo! Dataset has been curated and was ready to be used. The EMCA Data had to be prepared. As seen in Table 1, the volume of the data exceeds 500 Gbytes. Since the data was in the *json* format, it was parsed and pushed to the MongoDB [18]. Data were split into fourteen collections, each corresponding to a single day. Moreover, since individual *analytical groups* were identifiable, for this research, a representative group could be chosen. Selected groups had to meet the following requirements: (1) group is well-known to the data operator, to be able to validate the results; (2) group should be composed of all of three types of nodes to represent a realistic dataset; (3) group has to be large enough to include multiple types of anomalies. After evaluation, the largest group among potential candidates has been selected.

Each node gathers information, like CPU and memory usage, data I/O, network I/O, etc. Each of them is described by multiple parameters. Hence, we ended with more than 4000 time series in the dataset.

First, time series with degenerate distribution were eliminated, as they do not carry information. Second, the unification of timestamps was conducted. Services, independently, report data every 5 minutes. Thus, data had to be "resampled" to the "the nearest 5 minutes". If for the same time series, two observations were resampled to the same timestamp, their average was calculated. Hence, each day delivered 288 observations (4032 unified timestamps for two weeks).

Finally, missing values were considered. All of the time series that had no values for at least one day

(288 consecutive timestamps) were removed. Other missing values, were considered to be the result of three factors. (1) Temporary (up to 10 consecutive minutes) connection loss (e.g. loss of Internet connection). Here, the majority of data was missing around midnight. This was not a periodic anomaly, but an issue related to data export. In this case, missing values were interpolated. (2) Scheduled maintenance, or a system reboot (according to EMCA, such events last up to 10 minutes). (3) Unscheduled maintenance, or system reboot. For situations (2) and (3) (representing $\sim 0.1\%$ of data), if data were missing for more than 10 minutes, it was replaced with 0. Finally, time series were scaled, by subtracting mean and dividing by standard deviation.

As a result, the final EMCA Dataset consisted of 2543 time series; each containing 4032 data instances. Note that, while over 1000 time series were dropped, the final dataset included information from *all* 71 devices.

Across the project, Python 3.6.7 was used. For data manipulation and extraction – Pandas, NumPy, and pyMongo were used. Neural networks have been realized using Keras wrapper for Tensorflow 2.0. For paper from Section 3, referenced codes were used.

4.2. Exploring the complete dataset

Encouraged by results obtained by the PCA, in smaller systems, we decided to apply it to the single group EMCA Dataset. However, it involved confronting some limitations. (1) Operating on "dynamic" data, latent representation of the dataset had to be often updated, resulting in massive data extraction, to supply the model. Hence, the usefulness of PCA became doubtful even for a single analytic group. (2) For high-dimensional data, it was hard to establish a proper number of principal components. For any fixed number of components, some time series were reconstructed poorly, while others were reconstructed well, causing inconsistency in reconstruction error and, consequently, questionable results. Note that changing the number of components resulted in "flipped correctness". Quality of some badly reconstructed series improved; while those reconstructed well, became erroneous. To investigate this further, a group of time series (standardized to $\mathcal{N}(0,1)$ distribution), without abnormal behavior (by our expertise) was selected. When the PCA was applied, for some time series, the mean absolute error (MAE) of reconstruction was of order 0.2, while for others it reached 0.8. Such a big variance in reconstruction should (in theory) indicate an anomaly, which contradicted our data selection procedure. Even so, it is possible that in the selected group of time series there was a "hidden anomaly", but such a scenario is relatively unlikely. (3) Finally, PCA was designed to work with stationary time series, while EMCA Dataset contains drift in mean and variance, including seasonal behavior. As a result, for the first few components (being a linear combination of all input data), the weight of contribution, for all input variables, was close to being equal. Thus, components were segmented based on some general trends, and not variability of each time series. This behavior is proven in a lemma found in [19]. In theory, for this type of time series, even a single combination of input time series can explain most of the data variability. Therefore, at least for the time being, PCA has been eliminated.

It was claimed (see, Section 3) that problems with the PCA can be remedied by the application of Autoencoders (AE), with mini-batch/online learning. AE's allow modeling nonlinear relations in non-stationary time series. Therefore, four AE variants were examined: AE and VAE with/without long-short term memory (LSTM) learning. Both encoders and decoders consisted of up to three hidden layers, with a standard dropout mechanism, used for regularization. Specifically, dropout was realized by omitting randomly chosen neural units during training. After a number of exploratory experiments, the final ("best") architecture consisted of 2 hidden layers with 300 and 100 nodes.

LSTM is used together with Autoencoders to catch the temporal properties of the analyzed time series. The use of Variational Autoencoders was supposed to lead to an improvement, by allowing the model to learn a compressed representation of input data and, also, to represent it as a probability distribution. By using Kullback-Leibler divergence [20] it should be possible to achieve continuity and completeness of the latent space, meaning that similar data instances, in the latent space, should also be similar in the original space.

Based on a large number of experiments the following conclusions were formulated. (1) AE without LSTM often results in a high variance of the reconstructed signal. (2) VAE without LSTM overgeneralizes the problem, leading to results "similar" for all of the time series. (3) Both AE and VAE, combined with the LSTM step, yielded better performance (by capturing the temporal behavior of the series). Here, results obtained for the *training* dataset pointed to potential anomalies. Moreover, some of them matched anomalies detected by static thresholding (method currently used by EMCA).

The "best overall" results were achieved by the AE-LSTM pair. Here, the VAE-LSTM achieved similar results, but it was *almost twice as time-consuming* to train as the AE-LSTM. Nevertheless, while the

reconstruction of the training dataset was good, pointing to some valid anomalous behavior, performance on the test dataset was poor, involving a high rate of false positives. Here, an important issue has to be raised. While a somewhat high rate of false positives may not sound bad in a research paper, it is highly undesirable in a real-world data center, where each anomaly-alarm requires expert involvement and increases costs.

Finally, reconstructing data from thousands of time series "together", considerably decreased contributions of "smaller anomalies". Hence, only anomalies with the total highest reconstruction errors were captured.

It became clear that the use of the AE/VAE+LSTM (as well as the PCA), for datasets of the type and size similar to the EMCA Dataset, is not likely to be successful. Let it be stress that it is *not* claimed that it is impossible. Rather, since this work is exploratory, and one of its goals is to investigate a wide spectrum of approaches and establish their pros and cons, attention was turned to models operating separately on each time series. While aware of the possible loss of ability to detect collective anomalies, we were curious about the multitude of potential methods (described in Section 3).

4.3. Results for separate time series

Here, let us recall that available approaches differ in their applicability. For instance, some try to capture the seasonality of time series, while others apply active thresholding, to capture point anomalies. Therefore, it became clear that it would be useful to pre-explore their "capabilities" for a dataset that (a) is tagged, and (b) somewhat similar to the EMCA Dataset. Hence, attention was turned to Yahoo! Dataset (see, Section 2). The methodological claim is that if a given method works well for the Yahoo! Dataset, it becomes a candidate for application to the EMCA Dataset. This is not to state that other methods may not succeed. Nevertheless, a stepping stone was needed, to point in the right direction. Therefore, let's summarize the results obtained for the Yahoo! Dataset.

Before proceeding, let us outline the proposed approach (Online AutoRegressive with eXogenous variables, OARX). Based on experiences, summarized in Sections 3 and 4.2, the decision to use a linear autoregressive model was made. Such models tend to achieve a low false-positive rate (see, [8]), which is very important (in large systems, in particular), to avoid unnecessary maintenance interventions. Model is trained using stochastic gradient descent (SGD, [21]), allowing stream data processing and "instantaneous" capturing of detected anomalies. Here, instantaneous means capability to execute 4000 "fit and predict operations" during 1 second, on a standard PC unit [1]. These results are obtained using sequential computing, while they can be optimized to run in parallel.

To supply the model with seasonal information (here, daily seasonality) an exogenous (seasonal) variable was introduced. As daily seasonality was assumed, an array of 288 values was created (initialized with ones) corresponding to each timestamp. During each "training day" its values were updated, by averaging with the actual values. Hence, for the seasonal time series, a smoothed pattern was obtained, and for stationary, a noisy signal.

Moreover, regularization is used to diminish the influence of insignificant variables, and better generalization. Take, as an example, a time series without any seasonal pattern. In this case, regularization should "push" weights of the seasonal term towards zero, and focus on auto-regressive terms. Finally, for automatic thresholding, Q-function [22] is applied. It allows detecting situations when, instead of an anomaly, a concept drift appears. Assuming the presence of a seasonal variable, the space complexity of OARX is given by $O(w + p + \omega)$, where w corresponds to the length of the estimated seasonal pattern, p indicates the number of autoregressive terms in the model, and ω is the size of the window used in the Q-function value estimation.

Based on the SoTA analysis (Section 3), five models were tested: Spectral Residuals, Luminol, DSPOT, Donut, and OARX. Their performance was assessed by standard measures: F1-score, precision, and recall.

Let us note that the distribution of anomalous points in the Yahoo! Dataset is not uniform. In fact, it has a heavy tail, i.e. most of the existing anomalies occur in only a few time series. Thus, the goal was to minimize the influence of "highly anomalous" elements. Hence, for each time series (separately), three performance measures were computed and averaged across the dataset. The results are presented in Table 3. The reason that the Donut is separated from other models, is that it requires a "warm start". Thus, the first 30% of data instances were used for "warm start training", and the remaining 70% for running Donut.

Finally, the obvious has to be stressed. Even though Yahoo! Dataset is tagged, here it was approached as raw data and the "anomaly tags" were used *only* to check the performance of the tested approaches.

For almost all subsets from the Yahoo! Dataset, OARX outperforms other state-of-the-art models (or shows similar scores). Taking into account the focus of the following work, crucial are scores obtained for the A1 Subset, as it contains the real-world data. Results

[1] Intel Core i7-8750H, 32 GB RAM

Model	A1 Subset			A2 Subset			A3 Subset			A4 Subset		
	F	Pr.	Rec.	F	Pr.	Rec.	F	Pr.	Rec.	F	Pr.	Rec.
SR	0.29	0.29	0.45	0.67	0.58	0.95	0.83	0.76	**0.97**	**0.72**	0.64	**0.95**
Luminol	0.30	0.32	0.46	0.58	0.48	0.85	0.60	0.57	0.76	0.46	0.42	0.67
DSPOT	0.38	**0.49**	0.42	0.72	0.73	0.75	0.42	0.61	0.43	0.23	0.36	0.30
OARX	**0.51**	0.48	**0.58**	**0.95**	**0.95**	**0.97**	**0.86**	**0.96**	0.83	0.66	**0.68**	0.74
Donut	0.35	0.39	0.59	0.47	0.38	0.79	0.33	0.29	0.50	0.29	0.26	0.45

Table 3. Summary of averaged performance scores of models tested on Yahoo! Dataset.

show that the OARX obtains the best F1-measure. Good precision, on the A1 Subset, has been achieved also by the DSPOT. This is, likely, related to the fact that it focuses primarily on point anomalies, which seem to dominate in the A1 Subset. Similarly, in the A2 Subset, with the majority of stable time series, OARX and DSPOT perform best. It is worth noticing that the SR model shows the highest metric scores on the last (A4) subset. This is because the SR is most resistant to sudden changes in the mean, or the variance. Mediocre overall performance has been shown by the Donut. Noted also that the main drawback of this approach, as well as other neural-network-based ones, is the time complexity. Specifically, experiments showed it being an order of magnitude slower to indicate anomalies when compared with other solutions (e.g. for the A1 subset, Donut training took a few minutes, vs. several seconds for the OARX). This is important, considering that the real-world data that will have to be confronted is substantially larger than the Yahoo! Dataset.

Overall, gathered results indicate that, for the Yahoo! Dataset, OARX was better than (or, at least, not worse than) other approaches. Part of the reason for the, overall, low scores (for all models) on the A1 Subset might be problems with capturing contextual anomalies, ongoing for a longer time (as only anomaly start is indicated). However, this also indicates that, in general, a lot more work is needed to improve anomaly detection in actual data from data centers.

Nevertheless, since OARXis "almost the best currently available", it is worth trying in an actual RCA process. To validate this decision, an additional check was performed. By investigating the mean average percentage error (MAPE), we received information about its predicting performance. Across whole dataset, error was oscillating at MAPE $= 14\%$, for predicting the next timestamp value, which is reasonable.

4.4. Proposed RCA approach

Let us now describe, in full, the proposed approach to RCA, as applied to the EMCA Dataset. On the meta-level, RCA is described in standard literature (see [23]). However, a detailed formulation of individual steps depends on the application area. While potential methods that can be used for the "data center type problems", have been proposed, e.g. in [17] and [5], they are not without limitations. First is dependence on correlation, as a resource/node relation indicator. In large systems, with thousands of time series, this approach is very likely to indicate spurious relations (see, also [24]). Second, it was developed for much smaller systems and is PCA-based. Since it was established that PCA is not likely to succeed in the considered problem, this method was rejected. As a result, upon reflection on the proposals found in the literature, and on the results of our investigations (reported above), a complete RCA method was formulated. It consists of the following three steps:

1. Anomaly detection applied individually to each time series.
2. Clustering of anomalies, based on time of their occurrence.
3. Casual inferring among variables in clusters.

As the main challenge underlining the process is proper anomaly detection, based on the above results, the OARX was selected for the first step of the RCA.

For clustering, research presented in [25] is acknowledged. Casual inference models, executed on large datasets, often lead to a high false-positive rate. To avoid this problem, clustering of time series, based on the anomaly occurrence time is proposed. Thus, the correlation matrix is computed, using all prediction errors for each time series, obtained from the OARX. As a large number of clustering algorithms exist, one had to be selected. Since the objective is to apply Pearson correlation coefficient, algorithms like k-means were rejected (k-means is minimizing least-squares, not distances). Overall, hierarchical agglomerative clustering was used, due to its ease of application, and the ability to produce reorderings for the similarity matrix visualization, which can be used as a validation step. For future work, other distance-based algorithms, such as DBSCAN and its variants, can be experimented with. Overall, hierarchical clustering [26] is used to gather parameters from resources, where there is a possibility for co-influence. By analyzing the obtained dendrogram (in Figure 1), and multiple cluster sizes, a

division into 35 clusters was performed. Each cluster consisted of 8-70 variables, except for one cluster that included all of the time series that haven't had any anomalies, or had singular anomalies that haven't affected other nodes.

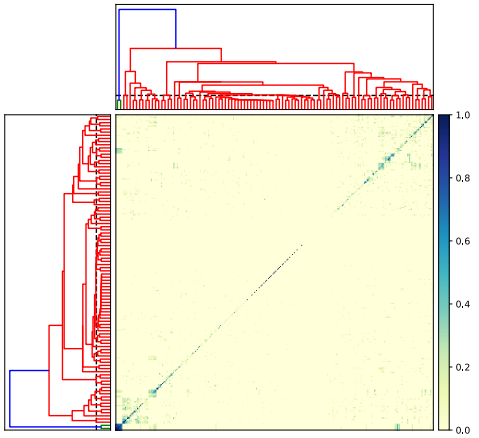

Figure 1. Dendrogram for analysis of co-occurring anomalies. Dashed line indicates pruning threshold.

At this point, time series have been clustered. The final step of the RCA is to construct a causality graph. However, based on [27], this task involves many difficulties, like the hidden confounder discovery, or handling non-stationary time series. Taking into account all these aspects, the PCMCI [25] has been selected. It consists of two steps: (1) Peter-Clarke algorithm for finding potential parents of each parameter indication, and (2) momentary conditional independence tests. As PCMCI assumes stationarity of the time series, they are differentiated before application (for each timestamp, previous data instance is subtracted from the current).

It is worth stressing that anomaly detection was performed independently for each time series. Hence, information concerning "what is happening in other time series" was not available. Therefore, the anomaly indication was *independent* from one of the time series to another. Nevertheless, after applying the OARX, interesting results have been observed.

- 10 out of 34 clusters were homogeneous (only parameters from a single node were included).
- Despite large cluster sizes (up to 70 parameters), most of them contain parameters from only up to 5 different nodes.
- One cluster contained a node connected to many other nodes (a hub). Thus, node degree distribution was similar to power-law (scale-free, see [28]) representing real-world scenarios. This hub (potential bottleneck) turned out to be *Disk Logical F: Current Queue Length*. In other words, a "weak point" was discovered (potential Root Cause of anomalies) that EMCA was not aware of, and which can affect system performance.

Selected causality graphs, in Figures 2, 3, 4, represent relations among parameters. Note that parameters were omitted when they did not involve a causal relationship with other variables. In the figures, color indicates the "origin resource", from which "parameters stem". For readability, directed arrows represent causal relations. Here, effects were observed within up to ten minutes (two timestamps) delay.

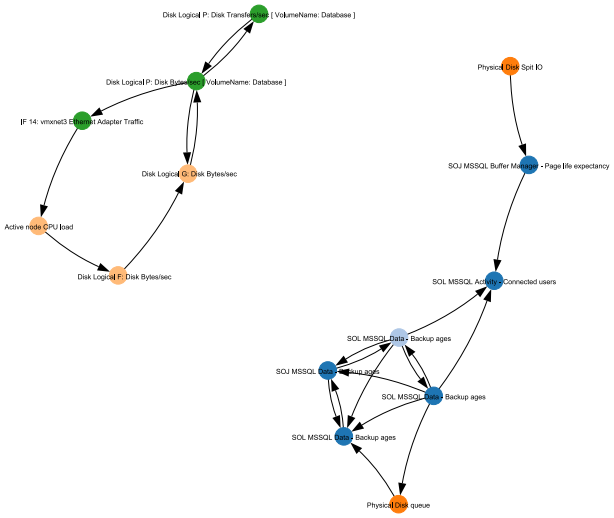

Figure 2. Causality graph for cluster no. 13.

Figure 2 shows that cluster no. 13 was separated into two components. In the"left one" direction of arrows suggests that the metric responsible for the disk subsystem performance influences other disk subsystems, causing performance degradation of underlying disks. On the "right-hand side", degradation of services, caused by bad disk subsystem performance, can be observed. Here, various database metrics were degraded because of the critical state of the disk queue.

Figure 3 shows a more complex behavior. The "influencers" are the CPU Usage and the free disk space. During the ten minutes interval, numerous MSSQL database metrics got degraded.

Finally, in Figure 4, "network connection failures", materialize and can be mapped into a problem of the Network interface IF12. If that becomes an actual bottleneck, a high CPU load might occur, with further influence on the performance of the delivered processes.

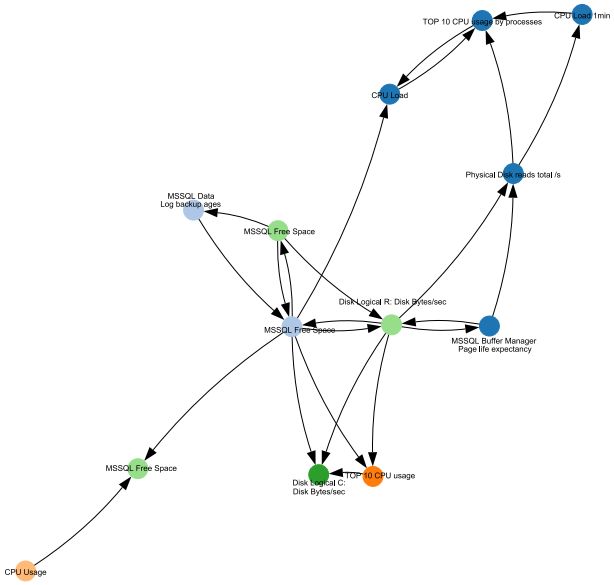

Figure 3. Causality graph for cluster no. 28.

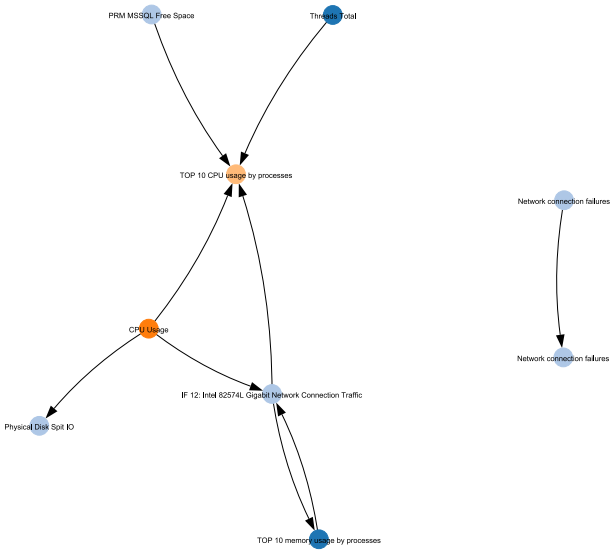

Figure 4. Causality graph for cluster no. 3.

5. Concluding remarks

In this study, progress in developing a comprehensive solution for RCA in data centers was reported. The proposed approach is based on a novel anomaly detection model, which is suited for streaming data, accounting for different types of anomalies, and applicable to raw data. Based on literature analysis, and performed experiments, reconstruction models (PCA and (V)AE) were rejected, and focus was turned into models operating on individual time series. Based on the state-of-the-art approaches, and an experimental search involving multiple approaches the OARX was proposed. Experiments showed that it outperforms other models when applied to the Yahoo! Dataset.

After anomalies were detected, they were clustered, based on the time of their occurrence. Afterward, to construct causal graphs, the PCMCI was applied. These graphs were interpreted by the EMCA experts and found to match actual/potential problems in the data center. Hence, the initial approach has been successfully validated in real-world settings.

However, despite achieved results, there is a huge room for improvement. (1) Even the best anomaly detection methods, applied to the A1 subset of Yahoo! Dataset (treated as raw data), should be improved. (2) There is no simple way of applying (semi-)supervised learning. As the OARX can spot point and contextual anomalies, the following approach can be envisioned. Assume that anomalies captured by the OARX, are human-verified (reducing time-effort, due to their expected reliability), resulting in a tagged dataset, ready for semi-supervised learning, leading to iterative model development. Another anchor point is the ability for causal inference models to operate on non-stationary time series, as currently employed differentiation might involve bias. Finally, historical causality graphs can be applied for future failure prediction. In upcoming research, it is planned to explore these, and other, avenues for developing a real-world RCA tool-set for data centers.

References

[1] C. M. Olszak and J. Zurada, "Big data in capturing business value," *Information Systems Management*, vol. 37, no. 3, pp. 240–254, 2020.

[2] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

[3] A. Blázquez-García, A. Conde, U. Mori, and J. A. Lozano, "A review on outlier/anomaly

detection in time series data," *arXiv preprint arXiv:2002.04236*, 2020.

[4] "S5 - a labeled anomaly detection dataset." `https://webscope.sandbox.yahoo.com`. Accessed: 2020-03-20.

[5] G. Li, J. Qin, and T. Yuan, "Data-driven root cause diagnosis of faults in process industries," *Chemometrics and Intelligent Laboratory Systems*, vol. 159, 09 2016.

[6] M. Sakurada and T. Yairi, "Anomaly detection using autoencoders with nonlinear dimensionality reduction," in *Proceedings of the MLSDA 2014*, pp. 4–11, 2014.

[7] Q. P. Nguyen, K. W. Lim, D. M. Divakaran, K. H. Low, and M. C. Chan, "GEE: A gradient-based explainable variational autoencoder for network anomaly detection," *CoRR*, 2019.

[8] F. Schmidt, F. Suri-Payer, A. Gulenko, M. Wallschläger, A. Acker, and O. Kao, "Unsupervised anomaly event detection for cloud monitoring using online arima," in *2018 IEEE/ACM Int. Conf. on UCC Companion*, pp. 71–76, IEEE, 2018.

[9] H. Ren, B. Xu, Y. Wang, C. Yi, C. Huang, X. Kou, T. Xing, M. Yang, J. Tong, and Q. Zhang, "Time-series anomaly detection service at microsoft," in *Proceedings of the 25th ACM SIGKDD*, pp. 3009–3017, 2019.

[10] M. Jacob, C. Neves, and D. Vukadinović, *Extreme Value Theory*, pp. 39–60. Cham: Springer International Publishing, 2020.

[11] A. Siffer, P.-A. Fouque, A. Termier, and C. Largouet, "Anomaly detection in streams with extreme value theory," in *Proceedings of the 23rd ACM SIGKDD*, pp. 1067–1075, 2017.

[12] J. Lechner, E. Simiu, and N. Heckert, "Assessment of 'peaks over threshold' methods for estimating extreme value distribution tails," *Structural Safety*, vol. 12, no. 4, pp. 305 – 314, 1993.

[13] M. Munir, S. A. Siddiqui, A. Dengel, and S. Ahmed, "Deepant: A deep learning approach for unsupervised anomaly detection in time series," *IEEE Access*, vol. 7, pp. 1991–2005, 2019.

[14] H. Xu, W. Chen, N. Zhao, Z. Li, J. Bu, *et al.*, "Unsupervised anomaly detection via vae for seasonal kpis in web applications," in *Proceedings of the 2018 WWW Conference*, pp. 187–196, 2018.

[15] R.-Q. Chen, G.-H. Shi, W.-L. Zhao, and C.-H. Liang, "Sequential vae-lstm for anomaly detection on time series," *arXiv preprint arXiv:1910.03818*, 2019.

[16] L. Wei, N. Kumar, V. Lolla, E. Keogh, S. Lonardi, and C. Ratanamahatana, "Assumption-free anomaly detection in time series.," in *In Proc. of the 17th Intl. Conf. on Scientific and Statistical Database Management (SSDBM), 2005*, pp. 237–240, 01 2005.

[17] LinkedIn, "Anomaly Detection library." `https://github.com/linkedin/luminol`, 2015. [Online; accessed 2020-04-01].

[18] K. Chodorow and M. Dirolf, *MongoDB: The Definitive Guide*. O'Reilly Media, Inc., 1st ed., 2010.

[19] J. Lansangan and E. Barrios, "Principal components analysis of nonstationary time series data," *Statistics and Computing*, vol. 19, pp. 173–187, 06 2009.

[20] F. Pérez-Cruz, "Kullback-leibler divergence estimation of continuous distributions," in *2008 IEEE international symposium on information theory*, pp. 1666–1670, IEEE, 2008.

[21] H. E. Robbins, "A stochastic approximation method," *Annals of Mathematical Statistics*, vol. 22, pp. 400–407, 2007.

[22] G. K. Karagiannidis and A. S. Lioumpas, "An improved approximation for the gaussian q-function," *IEEE Communications Letters*, vol. 11, no. 8, pp. 644–646, 2007.

[23] D. Okes, *Root Cause Analysis: The Core of Problem Solving and Corrective Action*. ASQ Quality Press, 2009.

[24] B. Haig, *Spurious correlation*, pp. 937–940. Sage Publications, Inc., 01 2007.

[25] J. Runge, "Causal network reconstruction from time series: From theoretical assumptions to practical estimation," *Chaos: An Interdisciplinary Journal of Nonlinear Science*, vol. 28, p. 75310, Jul 2018.

[26] F. Murtagh and P. Contreras, "Methods of hierarchical clustering," *Computing Research Repository - CORR*, 04 2011.

[27] M. Nauta, D. Bucur, and C. Seifert, "Causal discovery with attention-based convolutional neural networks," *Machine Learning and Knowledge Extraction*, pp. 312–340, 2019.

[28] R. Albert and A.-L. Barabási, "Statistical mechanics of complex networks," *arXiv preprint cond-mat/0106096*, 2001.