Communication

Not peer-reviewed version

A Study on Automated Problem Troubleshooting in Cloud Environments with Rule Induction and Verification

Arnak Poghosyan *, Ashot Harutyunyan *, Edgar Davtyan , Karen Petrosyan , Nelson Baloian

Posted Date: 8 December 2023

doi: 10.20944/preprints202312.0554.v1

Keywords: automated troubleshooting; real-time product activity detection; problem root cause analysis; machine learning; explainable AI; proactive SaaS support

Communication

A Study on Automated Problem Troubleshooting in Cloud Environments with Rule Induction and Verification

Arnak Poghosyan [1,2,*], Ashot Harutyunyan [1,3,4,*], Edgar Davtyan [5], Karen Petrosyan [6] and Nelson Baloian [7]

[1] VMware Inc
[2] Institute of Mathematics NAS RA
[3] Yerevan State University
[4] Institute for Informatics and Automation Problems NAS RA; {apoghosyan;aharutyunyan}@vmware.com
[5] Picsart
[6] American University of Armenia
[7] Department of Computer Science, University of Chile; edgar.davtyan@picsart.com, karen_petrosyan2@edu.aua.am, baloian@dcc.uchile.cl
* Correspondence: arnak@instmath.sci.am (A.P.); harutyunyan.ashot@ysu.am (A.H.)

Abstract: Remediation of IT issues encoded into domain-specific or user-defined alerts occurring in cloud environments and customer ecosystems in a vast majority of cases suffers from accurate recommendations which could be timely supplied for recovery of performance degradations. That, of course, is hard to realize through furnishing those abnormality definitions with an appropriate expert knowledge which varies from one environment to another. At the same time, in a large proportion of support cases, the reported problems under Global Support Services (GSS) or Site Reliability Engineering (SRE) treatment ultimately go down to the product teams making them waste costly development hours on investigating self-monitoring metrics of our solutions. Therefore, the mean-time-to-resolution (MTTR) rates of problems/alerts are significantly impacted from lack of a systematic approach towards adopting AI Ops. That would imply building, maintaining, and continuously improving/annotating a data store of insights that ML models are trained and generalized across the whole customer base and corporate cloud services. Our ongoing study is in line with such a vision and validates an approach that learns the alert resolution patterns in such a global setting and explains them using interpretable AI methodologies. The knowledge store of causative rules is then applied in predicting potential sources of the application degradation reflected in an active alert instance. In this communication, we share our experiences with a prototype solution and up to date analysis demonstrating how root conditions are discovered with a high accuracy for a specific type of problem. It is validated against the historical data of resolutions performed by heavy manual development efforts. We also offer a Dempster-Shafer theory-based rule verification framework for experts as a what-if analysis tool to test their hypotheses about underlying environment.

Keywords: automated troubleshooting; real-time product activity detection; problem root cause analysis; machine learning; explainable AI; proactive SaaS support

1. Introduction

Problem/Alert troubleshooting or root cause analysis (RCA) in cloud services remains a permanent headache for product engineers despite the extensive efforts, developed concepts, and toolsets, including Aria management platform [1–4] authors working on towards its enhancement with AI Ops capabilities.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 8 December 2023 doi:10.20944/preprints202312.0554.v1

Here is a typical chain (Figure 1) of activities that a product engineer faces across his/her troubleshooting journeys:

a. Aria Ops (former vR Ops) generates a mission-critical alert, the customer/user is not able to diagnose or even understand the situation.
b. SRE/GSS teams involve into the issue resolution.
c. If the issue necessitates, development teams are included in the process.
d. Development spends hours and days to perform root cause analysis.
e. It provides the fix of the problem.
f. Participating engineers gain valuable domain knowledge/expertise.

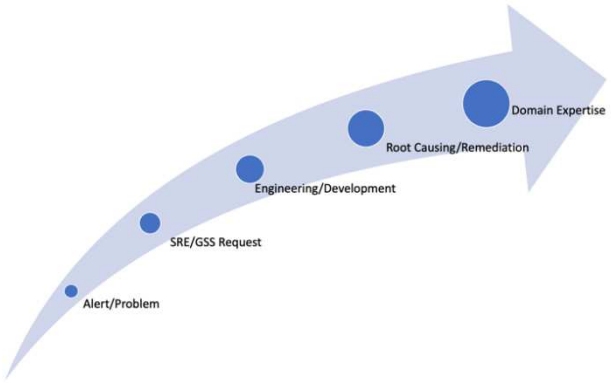

Figure 1. Traditional flow in a problem resolution.

Overall, this chain consumes a lot of time and professional resources and implies impacts on customer business applications loyalty.

At the same time, because of such intensive efforts, the knowledge gained during this process largely remains unsystematized for faster and more machine-based facilitation of handling the next cases to save critically important development resources and relocating them to the delivery of the core solutions/features in the roadmap of releases. Of course, in specific cases, when domain expertise is well developed and available, just checking some performance metrics (time series) and log messages it is quite straightforward to provide the fix of the problem with such a manual RCA. However, from the product perspectives, the development teams have no mechanism for automatic knowledge sharing and RCA, which eventually could help the customer to quickly resolve the issue without involvement of 3rd parties.

Our proposal is motivated by the above-mentioned traditional and outdated ways of problem troubleshooting and addresses the related lack of capabilities with a recommendation engine (ProbRCA) which is AI-driven and explainable for human operators. Users can easily approve/decline those recommendations for actions framework and/or enrich related alert definitions with resolution recipes.

This paper focuses on an explainable AI Ops [5] (see also survey [6] on RCA methods) approach to automatically identify conditions that recommend about the roots of a specific type of problem occurring across the customer base. Thus, for trending issues within the provider services, separate ML models need to be trained and continuously improved with additional factual data. This will allow providers to timely and even automatically fix the problems within a global analytics service/recommender system, especially when SaaS delivery model is concerned (with related opportunity to generalize cross-customer patterns based on available self-monitoring metrics of the cloud product).

We also incorporate into our study a rule validation mechanism based on Dempster-Shafer theory (DST) of evidence [7] with uncertainty modelling which essentially provides a “what-if” analysis framework for diagnosing the underlying system and its phenomena. This allows cloud users or developers of services to rigorously verify their hypotheses about the system behavior using

 doi:10.20944/preprints202312.0554.v1

a recently proposed interpretable classifier [8] with expert encoded rule conditions on system features. It enables owners/experts of the cloud infrastructure or application services to build a troubleshooting knowledge base that systemizes their professional wisdom with scientifically grounded theory and explainable ML mechanisms for more effective and automated diagnosis of business-critical software they are responsible for healthy availability.

2. Related Art

As noticed in Introduction, real-time diagnostics of offered services remains a challenge. Assistive frameworks, like Troubleshooting Workbench (TW) by Aria Operations [1], intelligent event/alert consolidation ways of guessing the origin of issues target this critical task from different angles. TW relies on discovery of important and relevant changes occurring within a delta time-and-topology-scope of the cloud infrastructure hierarchy that might evidence about the source of cause, while the alert grouping concepts rely on helping users to focus on the larger incidents to make inference of the root issues easier, as well as look into the problem in relevance with co-occurring events across time and infra/app topology axes. However, all these methods are inherently limited in automation capacities and intelligence power to perform deep and targeted RCA for alerts.

Other event management vendors, like Big Panda [9], Moogsoft [10], and Pager Duty [11] have adopted the consolidated insight and incident discovery strategy, but also build a vision on human-driven guidance of alerts and incidents consisting of those atomic events for training supervised RCA models. An important related art in the industry represents InfoSight [12] by HPE, an AI-powered autonomous operations service applying analytics from global learning with a self-managing, self-healing, and self-optimizing vision for cloud applications. In this regard, ProbRCA was proposed to realize a self-support for cloud management offerings with explainable features. It applies cross-customer user and developer feedback and trains accurate models over time to utilize them in recommending problem resolutions while also interpreting/justifying those measures.

3. Materials and Methods for ProbRCA

As an automation solution to the resource-expensive issue of managing problems in product troubleshooting, we suggest ProbRCA, an analytics system with AI Ops that builds and maintains ML models capable learning explainable and causative patterns (remediation rules) for alert/problem types. In Aria Ops integration scenario, it can automatically check the existence of those rules and proactively provide appropriate recommendations for resolution of a problem which is not even reported yet or reflected in the alerts stream.

As a result, user gets enough understanding of the problem and possible fix in a short time and without including additional resources and escaping time-intensive investments. ProbRCA essentially supports the pipeline in Figure 2 with the building blocks summarized in the following items (reflected in the process diagram of Figure 3):

a. Aira Ops generates mission-critical alerts.
b. ProbRCA sets its general scope to the alert-related other key performance indicators (KPIs) impacted and their monitoring data for trainings.
c. Related time series metrics preprocessing, e.g., smoothing, min/max normalization.
d. Executing rule induction learning.
e. Discovered rules are added to the library of rules.
f. Relevant rules are tracked and recommended for alert resolution.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 8 December 2023 doi:10.20944/preprints202312.0554.v1

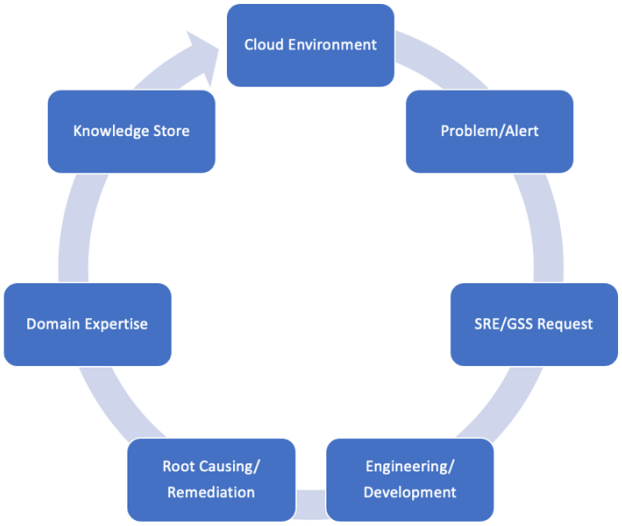

Figure 2. Proposed system (ProbRCA) with AI Ops.

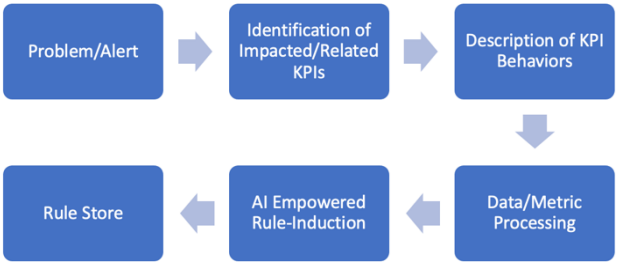

Figure 3. AI empowered rule-induction.

Within the SaaS offering model, the self-monitoring metrics for Aria Ops components are owned and managed by the provider, which means that there is a huge and extremely valuable datasets from different customers to be utilized for training the ML models underlying ProbRCA and delivering real-time troubleshooting.

This also means that even without the most advanced AI-driven intelligence, ProbRCA may still represent a problem solving, a common knowledge sharing/extraction system leveraging cross-customer insights, and a troubleshooting center relying on the basic conceptual and architectural components as depicted in Figure 4, where reactive and proactive problem resolution depends on the availability of relevant ML models trained on the data sets identified according to above mentioned items and the pipeline in Figure 3.

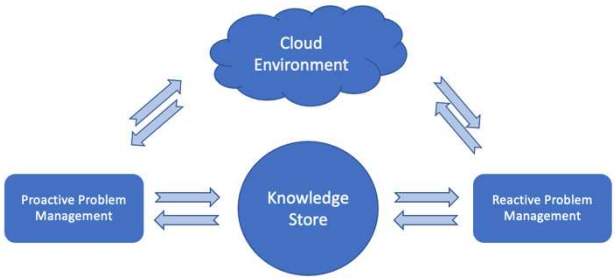

Figure 4. Knowledge-based proactive/reactive problem resolution.

Analysis of specific issues by developers at a serviced cloud eco-system with Aria Ops management solution and the objective of finding their root causes have resulted in interesting lessons learnt:

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 8 December 2023 doi:10.20944/preprints202312.0554.v1

- Aria Ops is collecting and storing data with some monitoring intervals. Traditional monitoring interval is 5 minutes. It means that Aria Ops is averaging the available values of a metric within this interval and storing them with a time stamp corresponding to the end of that interval. As a result, the average can vary from the actual value corresponding to that specific time stamp and the difference may be very large, especially in case of many outliers. We noticed that due to those random fluctuations some correlated metrics are no longer detectable by the correlation analytics in Aria Ops based on Pearson coefficient. Hence, this effect makes the correlation engine of the product unfairly useless.
- This synchronization problem can be only resolved by application of proper data-smoothing techniques. In our experiments, we apply a well-known min-max smoothing technique that Aria Ops is using in UI for visualizing many data points on a small window. It takes a time window (say 6 hours), finds the minimum and maximum values of a metric, and put them in the middle and at the end of that interval in the same order as they appeared in that interval. For example, if the minimum came earlier than the maximum, then the value of the minimum should be put in the middle and the value of the maximum in the end. Then, it shifts the window by 6 hours and reiterate the procedure until the end of the metric.
- Aria Ops is collecting a vast number of metrics from cloud infrastructures, but also a bunch of self-generated metrics constructed by domain experts. The final monitored datasets contain thousands of metrics with highly correlated subsets describing the same process. As a result, the metric correlation engine, or TW can detect hundreds of other metrics with the same behavior. But it will be very hard to separate the metrics that describe the same process from the metrics related to different ones for the detection of possible causations. This problem can be resolved only by users with some expertise. They need to manually separate the possible domains of interrelations and skip analysis within the same areas. That is why, we work below separately with three different datasets thus manually decreasing the total number of possible correlations.
- Alert/alarms are another source of uncertainty in Aria Ops. Many alerts are not directly connected to a problem as user-defined ones are not always sufficiently indicative. From the other side, many problems have appeared without a proper alert generation. In our example below, the problem is not connected to a known alert – 'Remote Collector Down' metric doesn't trigger any alert due to its fast oscillations. We found that a problem analysis is always starts from the corresponding KPI and its behavior. Even if the description of a problem is starting from an alert, the set of appropriate KPI metrics should be identified and described.
- Finally, as we mentioned before, the expert knowledge of Aria Ops engineers remains hidden in internal departments among a few numbers of specialists. As a rule, this knowledge is not systemized, not shared appropriately, cannot be used for a consistent and proactive management and a fast resolution of similar issues especially in cross-customer mode.

4. Trending Problem Scenarios

Let us describe a specific trending problem for a period which has impacted many user environments:

Customers didn't configure their firewall for Aria Ops Cloud Proxy [13] properly, it means ensuring firewalls are configured to allow outgoing traffic to:

⇒ *.vmwareidentity.com

⇒ gaz.csp-vidm-prod.com

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 8 December 2023 doi:10.20944/preprints202312.0554.v1

⇒ *.vmware.com

⇒ *.vrops-cloud.com

⇒ s3-us-west-2.amazonaws.com/vrops-cloud-proxy.

vR Ops Cloud Proxy is a primary component for data collection. In case of the SaaS offering, it is the only appliance deployed in the customer's environment, and only way for data collection. Because of that, over time a big set of different functionalities were added to it. All that means that whenever there is a problem with Cloud Proxy-to-vR Ops cluster communication, the data collection, alerts stream, etc. are all stopping to be serviced, in other words, vR Ops actually is not available.

Cloud Proxy basically contains two major services, collector service, which is responsible for the data collection from the endpoints and sending them to the vR Ops cluster, and CaSA (cluster and slice administration), which is responsible for the management of the Cloud Proxy, i.e. initial deployment, upgrade, configuration, etc. For both, as well as for the CP VM, vR Ops is collecting self-monitoring metrics. As displayed in Figure 3, all the communication from Cloud Proxy to vR Ops cluster is going out from HAproxy.

From the experience of engineers, whenever there is a problem on the customer side, e.g., deployment was not done properly, or network and firewall were not configured properly, it requires tremendous efforts to root cause the situation and validate the resolution.

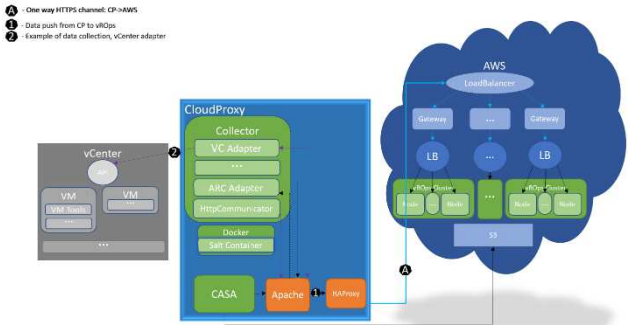

Figure 5. Cloud Proxy to vR Ops Cluster communication.

That is true especially in the cases related to the firewall, as it is requiring involvement of different departments and 3rd parties, customer's network team, security team, firewall support team, development. That causes an exhausting communication back and forth and time-consuming activities. We omit listing here SRs which evidence this and contain private information.

In all these cases the situation is rather simple. It is taking time until a customer is noticing the issue. Then GSS is getting involved, later developers join the investigations, and then, finally, it is becoming clear that the cause is the customer's firewall configuration. In the best case it is taking a week to figure it out and fix, but there were cases when it took longer. Also, it is requiring a lot of efforts to prove and convince the customer that problem is not vR Ops-related. This is a common story in practice.

Furthermore, in all these cases, Cloud Proxy goes down periodically. And the cloud proxies deployed within the same network and firewall, are acting with the same periodicity. Another observation is that in case of a product bug also, one service can have issues while sending the data to vR Ops cluster.

In this story, the developer's investigations lead to discovery of a common sequence of patterns:

- an issue with the collector service in the cloud proxy reported,
- CaSA service is not sending self-metrics as well,
- but whenever the cluster starts receiving (self-monitoring) metrics data, it turns out that the cloud proxy VM was not down during the span of the issue,
- these patterns are happening periodically and synchronously.

It is then becoming clear that the problem is network/firewall related.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 8 December 2023 doi:10.20944/preprints202312.0554.v1

We collected data sets on the problem instances on cloud proxy failures detailed above to train and validate interpretable ML algorithms capable for discovery the conditions or causes of those failures (which are already established by the developers as the ground truth of a such mis-performance).

5. Experiments and Discussions

According to the system diagram of ProbRCA, we conducted initial research on the above-mentioned trending problem and related identification of impacted KPIs. The cloud proxy down status was chosen. We have identified the periodicity and the same behavior for the KPI metrics in case if cloud proxies sharing same network and firewall. Later we identified components that can be useful to include into our analysis with their self-monitoring metrics (i.e., collector, CaSA, and Cloud Proxy).

5.1. Data

We utilize three different datasets synchronized by the time stamps.

The first data set contains the metrics from CaSA. It has 36 metrics (columns) and 7530 metric values (rows) with 5-min monitoring interval. Some of the metrics are: “API Calls AvgResponseTime”, “FreePhysicalMemory“, “GarbageCollector PS Scavenge Collection Time”, “MaxHeapSize”, “System Attributes Original Total Alert Count“, etc.

The second one contains the metrics of the collector with 138 columns and 7530 rows (5-min monitoring interval). Some of the redundant metrics were removed by engineers. This dataset is composed of metrics like “Control To Collector Task Action Status Elapsed Time Summary”, “Controller To Collector Perform Action Tasks Receive”, ”Collector To Controller Get Adapter Ids Elapsed Time Summary”, etc.

The third dataset is a collection of cloud proxy metrics with 151 columns and 7530 rows (5-min monitoring interval). Some of the names of metrics are “Net TCP CP Close Wait”, “Data Receiving Status“, “Disk File System Storage DB Files Free“, “Disk File System Write Bytes”, “Net All Inbound Total”, etc.

5.2. Specific Results

According to our general planning, we start with the identification of the KPI. For this specific problem, the corresponding alert is missing but the “Remote Collector Down” KPI has a typical behavior shown in Figure 6. It started to oscillate going up and down rather frequently indicating a problem. The same behavior of the same KPI have been detected for a series of customers with totally independent cloud environments.

We apply min-max smoothing with 6-hour time interval before the application of more intelligent solutions. Figure 7 confirms the oscillating behavior of the KPI after the smoothing. It reduces the number of metric values from 7530 to 208.

We analyze connections between the KPI and other indicative metrics separately for CaSA, collector, and Cloud Proxy.

Since our research focuses on interpretable ML (see [14,15]) strategies, we apply the classification rule-induction system RIPPER (see the related literature [16,17]) which reveals rules containing the names of important metrics with some thresholds combined in conditions. The labeling of datasets is performed via the values of KPI (labels = 0,1) where label=1 indicates the “down” status of the cloud proxy. We have detected 44 down conditions from 208 available data points.

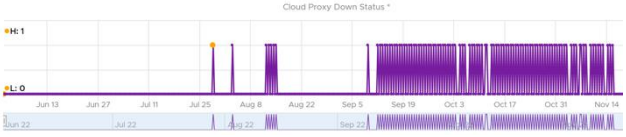

Figure 6. The oscillating behavior of “Remote Collector Down” KPI.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 8 December 2023 doi:10.20944/preprints202312.0554.v1

Applying RIPPER to the CaSA dataset exposed the following rule (Rule 1):

(CaSA API Calls Total Requests $\leq$ 0.1) and

(CaSA Garbage Collector Aggregated Collection Time $\geq$ 0.03) $\rightarrow$ KPI = Down (23/1).

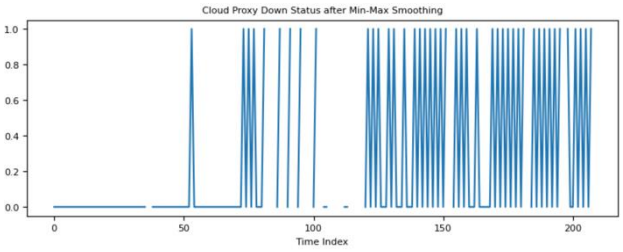

Figure 7. "Remote Collector Down" KPI metric after the min-max smoothing.

The fraction at the end of the rule characterizes the importance. The number 23 shows how many times the rule has been fired and denominator indicates the number of misclassifications. The coverage of the rule is

$$22/44 * 100\% = 0.5\%$$

and the accuracy is

$$22/23 * 100\% = 96\%.$$

Applying RIPPER to the collector dataset detects the second rule (Rule 2):

(Collector To Controller Lookup Resource Elapsed Time Summary $\geq$ 0.002) $\rightarrow$ KPI = Down (42/5)

This rule has 84% coverage and 88% confidence.

Applying RIPPER to the Cloud Proxy dataset returns the third rule (Rule 3):

(Net TCP Listen = 0) $\rightarrow$ KPI = Down (47/9)

with 86% coverage and 80% accuracy.

Those rules and the KPI's specific behavior can be stored as a knowledge that can be used for identification and fast resolution of similar issues. Aria Ops engineers have verified the Rules 1-3 ensuring that in combination they indicate the firewall problems.

5.3. Rule Validation with Dempster-Shafer Theory

To generalize expert views or knowledge gained over larger contexts and measured data horizons that relates to various scales of product eco-systems and type of workloads they manage, we alternatively study a special rule verification framework and the corresponding classifier proposed in [8]. It takes expert hypotheses as rules defined on features (and their combinations) to attest their quality thus realizing an interpretable what-if analysis for its further utilization in the knowledge store and real-time remedial executions. This framework utilizes DST of evidence or plausibility as an additional assistance to help experts validating their wisdoms about the environment conditions in the classification setting. Thus, if their experience tells them that some patterns or behaviors of specific features within particular ranges might lead to the system misbehave, they plug those hypotheses (in other words, rules) into the DS classifier and get validation of their quality including uncertainty estimates to account for. Moreover, the classical rule induction algorithms and DS rule verification approach can be leveraged combined while feeding the automatically learned rules from the first strategy to be estimated with the second. In particular, Rule 1 discovered by RIPPER gets a low uncertainty estimate through DS rule testing.

The user of this framework can simply break value ranges (rule intervals) of all the data frame features to verify how they can be indicative standalone or in any combination. Here is a set of rules validated with splitting features into two ranges, which in several cases surprisingly results in high quality rules:

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 8 December 2023 doi:10.20944/preprints202312.0554.v1

CaSA_APICalls_TotalRequests < 0.424,

with probability of positive class =0.998 and uncertainty =0.002

CaSA_Threads = 0.5,

with probability of positive class =0.993 and uncertainty =0.067

CaSA_FreePhysicalMemory > 0.397,

with probability of positive class =0.859 and uncertainty =0.141

CaSA_GarbageCollector_PS MarkSweep_collectionTime > 0.218,

with probability of positive class=0.857 and uncertainty = 0.143.

Another complex rule enforced by the user gets the following estimate:

CaSA_GarbageCollector_PS MarkSweep_collectionCount = 1 and

CaSA_GarbageCollector_PS Scavenge_collectionCount = 0,

with probability of positive class =0.973 and uncertainty =0.027.

In the final version of the paper, we'll be sharing more insights evaluated using DST.

6. Evaluation of Results

In our further evaluation analysis, we are applying available knowledge to an unknown issue. Aria Ops detected similar problems in other customer environments. We analyze one of those problems. The inspection of "Remote Collector Down" KPI showed the same behavior as in Figure 6 and 7. We detected 46 cases when the cloud proxy was down. Verification of rules showed that in case of Rule 1, the rule was correctly fired in 31 of cases. In case of Rule 2, the rule was correctly fired in 32 of cases. In case of Rule 3, the rule was correctly fired in 38 of cases. We can confirm the matching of the problem pattern stored in the library of rules. These highly accurate results on unseen data demonstrate the feasibility of the chosen approach and the quality of explainable ML models trained. Figures 8 and 9 present the behavior of some of the metrics participated in the rules in combination with "Remote Collector Down" KPI after min-max smoothing for better clarification of the correlations.

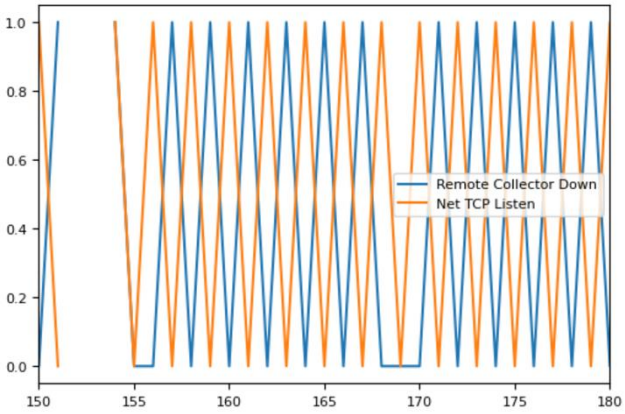

Figure 8. Behaviors of "Remote Collector Down" and "Net TCP Listen" metrics.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 8 December 2023 doi:10.20944/preprints202312.0554.v1

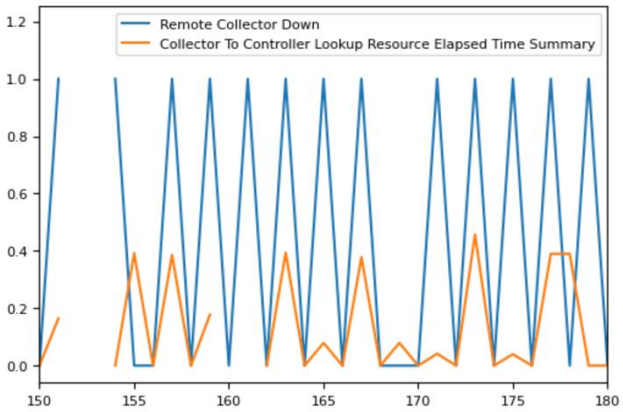

Figure 9. Behaviors of "Remote Collector Down" and "Collector to Controller …" metrics.

7. Conclusion and Future Work

By analyzing trending customer problems within cloud services, we validate an explainable ML method capable of learning underlying common patterns of the same type issues and pointing out conditions or sources of such anomalous behaviors. Machine detected causality conditions can enrich the alert definitions in cloud operations services thus enhancing their event management capabilities with an AI-driven problem resolution assistance as an easily reachable implication of this work. We also described our larger vision on building a real-time self-diagnostic system and RCA tool based on global learning across the customer eco-systems within the SaaS model of cloud delivery. Our method demonstrated accurate predictions of ground truth root causes of those problem types. We plan to extend our analysis to other problem types and derive insights and quality metrics from much larger experimental test beds.

Architecting ProbRCA service end-to-end requires substantial development work. Within this study our objective was to prove the viability of such a system from data science perspectives. We also plan to put efforts in adopting DST-based approaches for achieving a better understanding of frontiers of this theory and its practical significance as an apparatus for characterizing and comprehending managed services.

This communication relates to and builds upon our prior research [18–25] on various specific tasks in cloud diagnostics and administration comprising time series forecasting, anomaly and change detection in not only such structured data but also in logs and traces, as well as event correlation analytics and abnormality root cause inference from those types of information sources for a comprehensive automation towards self-driving data centers.

8. Patents

The study is supported by a filed (2023) US patent.

Funding: Authors research was supported by ADVANCE Research Grants provided by the Foundation for Armenian Science and Technology.

References

1. *VMware Aria Operations:* https://www.vmware.com/products/vrealize-operations.html.
2. *VMware Aria Operations for Applications:* https://www.vmware.com/products/aria-operations-for-applications.html.
3. VMware Aria Operations for Logs, https://www.vmware.com/products/vrealize-log-insight.
4. VMware Aria Operations for Networks, https://www.vmware.com/products/vrealize-network-insight.html.
5. *AI ops by Gartner,* https://www.gartner.com/en/information-technology/glossary/aiops-artificial-intelligence-operations.
6. *M. Sole, V. Muntes-Mulero, A.I. Rana, and G. Estrada, Survey on Models and Techniques for Root-Cause Analysis. arXiv: 1701.08556v2, 2017.*

7. *G. Shafer, A mathematical theory of evidence, Princeton University Press, 1976.*
8. S. Peñafiel, N. Baloian, H. Sanson, J.A. Pino, Applying Dempster–Shafer theory for developing a flexible, accurate and interpretable classifier, Expert Systems with Applications, vol. 148, 113262, 2020. https://doi.org/10.1016/j.eswa.2020.113262.
9. *Big Panda,* https://www.bigpanda.io/.
10. *Moogsoft,* https://www.moogsoft.com/.
11. *Pager Duty*, https://www.pagerduty.com/.
12. *HPE InfoSight,* https://www.hpe.com/us/en/solutions/infosight.html.
13. *Configuring VMware Cloud Proxies,* https://docs.vmware.com/en/vRealize-Operations/Cloud/getting-started/GUID-7C52B725-4675-4A58-A0AF-6246AEFA45CD.html.
14. *A.B. Arrieta, N. Díaz-Rodríguez, et al., Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI, Information Fusion, v. 58, pp. 82-115, 2020.*
15. M.T. Ribeira, S. Singh, and C. Guestrin, Why should I trust you?: Explaining the predictions of any classifier, https://arxiv.org/pdf/1602.04938v1.pdf, 2016.
16. *W. Cohen, Fast effective rule induction, Proceedings 12th International Conference on Machine Learning, Tahoe City, California, July 9–12, pp. 115-123, 1995.*
17. *J. Fürnkranz, D. Gamberger, N. Lavrac, Foundations of Rule Learning. Springer-Verlag, 2012.*
18. A. Poghosyan, A. Harutyunyan, N. Grigoryan, C. Pang., G. Oganesyan, S. Ghazaryan, N. Hovhannisyan., An enterprise time series forecasting system for cloud applications using transfer learning, Sensors 2021, 21(5): 1590. https://doi.org/10.3390/s21051590.
19. A.V. Poghosyan, A.N. Harutyunyan, N.M. Grigoryan, N. Kushmerick, Incident management for explainable and automated root cause analysis in cloud data centers. Journal of Universal Computer Science 27(11), pp. 1152-1173, 2021.
20. A.N. Harutyunyan, N.M. Grigoryan, and A.V. Poghosyan. 2020. Fingerprinting data center problems with association rules, Proceedings of 2nd International Workshop on Collaborative Technologies and Data Science in Artificial Intelligence Applications (CODASSCA 2020), September 14-17, American University of Armenia, Yerevan, Armenia, 159-168, 2020.
21. A.N. Harutyunyan, A.V. Poghosyan, N.M. Grigoryan, N.A. Hovhannisyan, N. Kushmerick, On machine learning approaches for automated log management, Journal of Universal Computer Science 25(8), pp. 925-945, 2019. https://doi.org/10.3217/jucs-025-08-0925
22. A. Harutyunyan, A. Poghosyan, N. Grigoryan, N. Kushmerick, and H. Beybutyan, Identifying changed or sick resources from logs, Proceedings of 2018 IEEE 3rd International Workshops on Foundations and Applications of Self* Systems (FAS*W), Trento, Italy, pp. 86-91, 2018. doi: 10.1109/FAS-W.2018.00030.
23. A.V. Poghosyan, A.N. Harutyunyan and N.M. Grigoryan, Compression for time series databases using independent and principal component analysis, Proceedings of 2017 IEEE International Conference on Autonomic Computing (ICAC), Columbus, OH, USA, 2017, pp. 279-284, doi: 10.1109/ICAC.2017.49.
24. A.V. Poghosyan, A.N. Harutyunyan, and N. M. Grigoryan, Managing cloud infrastructures by a multi-layer data analytics, Proceedings of 2016 IEEE International Conference on Autonomic Computing (ICAC), Wuerzburg, Germany, pp. 351-356, 2016. doi: 10.1109/ICAC.2016.33.
25. A.N. Harutyunyan, A.V. Poghosyan, N.M. Grigoryan, and M. Marvasti, Abnormality analysis of streamed log data, Proceedings of IEEE Network Operations and Management Symposium (NOMS 2014), May 5-9, Krakow, Poland, 2014.