HURRA! Human-Readable Router Anomaly Detection

José M. Navarro, Dario Rossi
Huawei Technologies Co. Ltd – first.last@huawei.com

***Abstract*—Automated troubleshooting tools must be based on solid and principled algorithms to be *useful*. However, these tools need to be easily accessible for non-experts, thus requiring to also be *usable*. This demo combines both requirements by combining an anomaly detection engine inspired by Auto-ML principles, that combines multiple methods to find robust solutions, with automated ranking of results to provide an intuitive interface that is remindful of a search engine. The net result is that HURRA! simplifies as much as possible human operators interaction while providing them with the most useful results first. In the demo, we contrast manual labeling of individual features gathered from human operators from real troubleshooting tickets with results returned by the engine – showing an empirically good match at a fraction of the human labor.**

I. Introduction

In the newspapers, Artificial Intelligence (AI) achievements are highly sensationalized, with projections of massive losses in the workforce due to increasingly automated operations. Part of it is surely true, since automation was already successful in increasing the amount of objects (e.g., servers, routers, etc.) that a single human can be responsible of. At the same time, a more likely outcome is that *AI will assist the human workforce*, significantly speeding up their operation and increasing system efficiency beyond human capabilities. However, whereas AI algorithms designers have the skills to understand the systems they create, it is clear that human operators interacting with such systems will only have a very loose knowledge of their inner workings, at best. As such, it is clear that human interaction with AI will become of increasing importance in the foreseable future [1]: this is true in many circumstances, from tasks such as autonomous driving, to the case of network troubleshooting – which is of paramount importance in current networks [2] and the focus of this demo.

In particular, when the underlying system's complexity increases (e.g. nowadays router can expose up to 70K metrics through YANG telemetry) manual investigation of such time-series is extremely time consuming and clearly error-prone. Our aim is to design a system that, while based on sound algorithmic principles, is also easy to interact with: to do so, we design the system considering the output from the point of view of a network expert, who has solid engineering skills and domain expertise but lacks PhD-level scientific skills and methodological/algorithmic expertise.

We showcase the system on two types of dataset, including real troubleshooting tickets from over 30 ISPs as well as testbed data where the fault injection process is under our control.

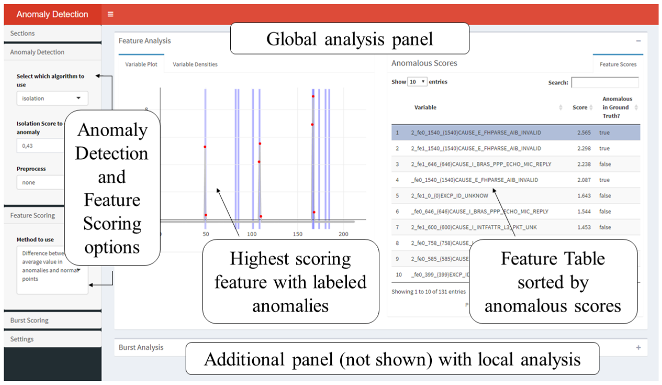
II. System design

Under this angle, it is clear that the most important resource human operators have is their time-budget: it follows that reducing the amount of time it takes humans to troubleshoot the problem, find a solution, file a report and close the case is the primary metric our system should be aimed at. Considering the first stage, i.e., troubleshooting , (i) one of the main challenges is to make algorithmic output *immediately understandable* by the operator; (ii) a second desirable goal is to rank output presenting *the most useful information first*, while (iii) keeping *interaction as simple as possible*.

We implement this vision in a system that, from a very-high level, can be thought of as offering a search engine interface for troubles: the system, which is aptly named HURRA!, makes it extremely easy for operators to quickly browse through hundreds of features. HURRA! decouples the temporal vs spatial angles, defining separate algorithms for the *temporal anomaly detection* (AD) problem, i.e., which timeslots to focus on in simple human terms, and for the *spatial feature scoring* (FS), i.e., providing a complementary mechanism that serves both as attention focus mechanism, as well as an human-readable explanation of anomalous features.

More formally, measuring F-dimensional feature vectors x over T time-slots, our dataset can be expressed as a $F \times T$ matrix for each node in the network. The application of multivariate AD methods finds anomalous moments in time in the dataset, i.e., a set of rows $AD \in \{1, ..., T\}$ which corresponds to the detected anomalous points of the dataset. To reduce cluttering, contiguous anomalies are aggregated, and anomalous events are sorted by decreasing importance (based on their magnitude). For each anomalous event, FS methods generate a set $\{s_1, s_2, ..., s_F\}$ of scores from which features can be ranked, prioritizing human attention to the most important feature first. In turn, this reduces the time taken by the expert, that would otherwise have to manually verify each features one-by-one (which is typically done nowadays).

Combining these two simple building blocks allows to focus in the timeslots where anomalies are (sorting them by relative level of importance when multiple are present), and presenting features in order of decreasing relative importance, as a search engine would do. Just like in these, advanced parameters are hidden but still accessible – so that while most of the operations are automated, humans can still take control over the algorithm settings to tackle corner cases (where e.g., default hyper-parametrization is failing).



Select which algorithm to use: isolation

Isolation Score to anomaly: 0,43

Preprocess: none

Method to use: Difference between average value in anomalies and normal points

Show 10 entries

	Variable	Score	Anomalous in Ground Truth?
1	2_fe0_1540_(1540)CAUSE_E_FHPARSE_AIB_INVALID	2.565	true
2	2_fe1_1540_(1540)CAUSE_E_FHPARSE_AIB_INVALID	2.298	true
3	2_fe1_646_(646)CAUSE_I_BRAS_PPP_ECHO_MIC_REPLY	2.238	false
4	_fe0_1540_(1540)CAUSE_E_FHPARSE_AIB_INVALID	2.087	true
5	2_fe1_0_(0)EXCP_ID_UNKNOW	1.643	false
6	_fe0_646_(646)CAUSE_I_BRAS_PPP_ECHO_MIC_REPLY	1.544	false
7	2_fe1_600_(600)CAUSE_I_INTFATTR_L3_PKT_UNK	1.453	false
8	2_fe0_758_(758)CAUSE_		
9	2_fe0_585_(585)CAUSE		
10	_fe0_399_(399)EXCP_ID		

Showing 1 to 10 of 131 entries

Global analysis panel

Anomaly Detection and Feature Scoring options

Highest scoring feature with labeled anomalies

Feature Table sorted by anomalous scores

Additional panel (not shown) with local analysis

Figure 1: Example output of the single-router datasets (annotated excerpt from the video available at [3].

III. Datasets

In the demo, the users can test HURRA! on both local (i.e., single-router telemetry) and global (i.e., network-wide telemetry) cases, using datasets with two kinds of ground truth.

Single-router telemetry For the local case, we resort to proprietary datasets, where we can access real troubleshooting tickets in over 30 operational networks. In this case, since anomalies are not injected, their root cause is not known a priori, but human operators label features according to their importance in solving the case, as well as the period where the features are found to be abnormal. This dataset allows us to especially quantify the *level of agreement between human and automated ranking*, which is a rather novel aspect and of special interest within our goals. Interestingly, these datasets are extremely diverse in temporal duration (from 211 to almost 11K rows), spatial diversity (from 6 to 373 features) and type/severity of anomalies (from punctual outliers affecting few features, to contextual anomalies that affect several features for a relatively long duration). A bird's eye view of this dataset variety is provided with the scatter plot in Fig. 2. We point out that the dataset is proprietary: while we plan to let participant interact with an anonymized version of the dataset during the demo session, we cannot make this data available in an online demo accessible from the Web. At submission time, we can only briefly expose the dataset via the video accompanying this submission accessible at [3].

Network-wide telemetry. For the global case, we focus on recently released publicly available dataset [4] where anomalies are purposely injected in a controlled data-center network testbed. In particular, BGP anomalies were induced by automatically triggering commands every five minutes in different network locations, so that the cause, location and time of the anomaly are known (but telemetry features are otherwise unlabeled). This dataset allows us to assess the validity of the anomaly detection algorithms, and to some extent, the relevance of the feature scoring system (since the root cause is in this case known and tied to the BGP protocol).

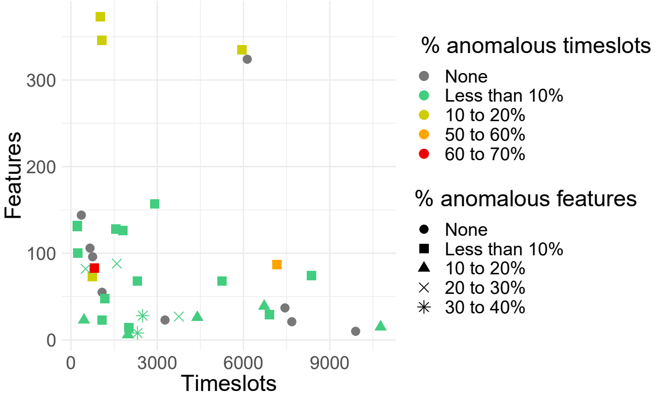


Figure 2: Single-router Datasets diversity at a glance (datasets are proprietary and are not available in the online demo, but will be available on site)

Given that this dataset is publicly available, we make the demo interface accessible online [3] at time of submission. Clearly, in this case as the labeling only record the start time of the anomaly, it is not possible to quantify the level of agreement between human and automated ranking (however in the online demo it can be seen that features related to BGP path count are the most relevant, which is coherent with the way the anomalies were generated).

IV. Demo workflow

In particular, users will be able to interact with the HURRA dashboard to perform from (i) simple and fully automated tasks, to (ii) more complex and interactive tasks. Simple automated tasks range from data exploration, to automated anomaly detection, e.g., where the system attempts at selecting the most suitable algorithm and hyper-parametrization. The intended target of this operation mode is clearly a human operator with extensive network domain expertise and limited AI skills. More complex interactive tasks include interacting with the inner AD and FS building blocks of the system. For instance, users can select the *anomaly detection algorithm* (such as Isolation Forest [5] for offline detection and DBStream [6] for online detection) and affect their hyper-parametrization. Similarly, demo users will be able to alter the *feature scoring policy* to, e.g., use only the current data, vs reuse expert knowledge from previous tickets to affect the feature ranking output.

References

[1] https://www.aiforhumanity.fr/en/.

[2] R. Domingues et al., "A comparative evaluation of outlier detection algorithms: Experiments and analyses," *Pattern Recognition*, vol. 74, pp. 406–421, 2018.

[3] https://huawei-prc-anomaly-detection.shinyapps.io/2020_infocom_demo/.

[4] https://github.com/anrputina/OutlierDenStream-BigDama18.

[5] F.Liu et al., "Isolation forest," in *IEEE ICDM*, 2008, pp. 413–422.

[6] M. Hahsler and M. Bolaños, "Clustering data streams based on shared density between micro-clusters," *IEEE Transactions on Knowledge and Data Engineering*, vol. 28, no. 6, pp. 1449–1461, 2016.