Towards a cloud-based framework for online and integrated event detection*

Janio Lima[1], Rebecca Salles[1], Luciana Escobar[1], Cristiane Géa[1,4], Pedro Alpis Fernandes[1], Esther Pacitti[2], Fabio Porto[3], Rafaelli Coutinho[1], Eduardo Ogasawara[1]

[1]CEFET/RJ - Centro Federal de Educação Tecnológica Celso Suckow da Fonseca

[2]INRIA & University of Montpellier

[3]LNCC - Laboratório Nacional de Computação Científica

[4]UFF - Universidade Federal Fluminense

{janio.lima, rebecca.salles, luciana.escobar}@aluno.cefet-rj.br

{cristiane.gea,pedro.alpis}@aluno.cefet-rj.br

esther.pacitti@lirmm.fr, fporto@lncc.br

rafaelli.coutinho@cefet-rj.br, eogasawara@ieee.org

Abstract. *Time series events detection relates to the study of techniques for detecting points in a series with special meaning which differs from the expected behavior of the dataset. In scenarios such as digital twins and IoT devices, there is natural generation and traffic of data in the cloud. Event detection is critical for timely decision-making. Since many methods for detecting events target different types selecting a suitable method makes the task more difficult. In this context, this article proposes a cloud-based framework called Harbinger Nimbus. The implementation was evaluated on the Microsoft Azure platform.*

1. Introduction

In the scenario of digital twins and IoT (internet of things) devices, it is possible to observe an increase in the speed of data generation in streaming in the cloud. In these scenarios, event detection becomes essential for timely decision-making and monitoring of the cloud infrastructure itself [Habeeb et al., 2019]. However, the event detection in streaming brings additional difficulties related to the change in data behavior over time and the need for adaptation [Talagala et al., 2020]. Furthermore, the importance of the response speed of the methods and how the integration of event detection influences this response time into data in streaming in the cloud increases.

In this work, events refer, for example, to anomalies and change points [Gensler and Sick, 2018]. There is a myriad of methods aimed at detecting events such as those cited in searches by Truong et al. [2020] for *offline* data or by Habeeb et al. [2019] for *online* data. However, there is still a shortage of works dedicated to *frameworks* for experimentation and integrating event detection methods focused on data streaming in the cloud.

*The authors thank FAPERJ, CAPES (code 001) and CNPq for funding the project.

In order to fill the gap in the literature, the present work proposes a framework for event detection in streaming data integrated into the cloud computing environment. Thus, *Harbinger Nimbus* (HN) is a framework for integrating a broad range of event detection methods in the cloud.

HN contributes to decision-making simplification, given its readiness for integration with other cloud applications. Finally, the HN prototype can be integrated with available methods implemented in the cloud, such as the *Microsoft Anomaly Detector* (MAD) native to *Azure* [Ren et al., 2019].

In addition to this introduction, the work presents in Section 2 a synthesis of works related to detecting events in time series. Section 3 presents the proposed architecture of HN. Section 4 makes final remarks.

2. Related Works

Event detection aims to identify points with special meaning in a time series. Although identified as specific points, they may be related to series intervals around the detected event with a significant change in the behavior of the data. This change refers, for example, to spikes, change points, or anomalies [Guralnik and Srivastava, 1999].

One way to divide the analysis of the series is in terms of data access. When access is in real-time the analysis is called (i) *online* detection. On the other hand, it is called (ii) *offline* detection when you have previous access to the complete dataset [Truong et al., 2020]. A comprehensive review of methods applicable to *offline* detection is presented by Truong et al. [2020]. However, this work focuses on *online* detection.

In the literature, several methods and *frameworks* apply to the detection of *online* events, with data generated in real-time. An overview of state of the art in this area is presented by Habeeb et al. [2019]. Besides, two main divisions in the detection of *online* events: (i) static models and (ii) models of incremental learning [Ogasawara et al., 2021]. Static models are trained on large datasets and applied to real-time data. In incremental learning models, training starts on a subset of data, and learning continues as new observations are received [Habeeb et al., 2019].

Harbinger is a framework that provides functions for event detection, evaluation and combination of methods, and comparison of the detections performed [Salles et al., 2020]. Another framework for *online* detection is proposed by Talagala et al. [2020], based on computing boundaries of normal data behavior to identify significant changes in new observations.

An anomaly detection method was introduced by Ren et al. [2019] as a service on the *Microsoft Azure* platform. This service, called MAD, is based on the *spectral residual (SR)* algorithm combined with convolutional neural networks (CNN) [Ren et al., 2019]. Another event detection service on a cloud platform is *Amazon QuickSight ML-powered anomaly detection*[1] from the AWS platform. However, no publications were identified with details about its operation or the method used by this service for comparison in this work.

The *streaming* is related to the traffic and processing of data in real-time, before

[1] Available at: https://docs.aws.amazon.com/quicksight/latest/user/anomaly- detection.html

its storage [Hiraman et al., 2018]. Habeeb et al. [2019] cite *Apache Kafka* and *Spark streaming* as examples of specific platforms for processing *streaming* data. These platforms allow real-time data exchange and processing of large volumes of data [Hiraman et al., 2018].

To the best of the authors' knowledge, there is a lack of *frameworks* that simultaneously allow the integration, comparison, and combination of different event detection methods in streaming data. Therefore, the HN proposes to explore this gap.

3. *Harbinger Nimbus*

As an evolution of the Harbinger research, the present work presents the Harbinger Nimbus (HN). The HN architecture is represented in Figure 1. The main focus of HN is to enable the integration of *Harbinger* in the cloud. It simplifies applying event detection methods in different data sources and applications that consume the detection result.

On the sides of the Figure 1 we have the connections with cloud data sources (1), intermediate by a *streaming* management system (2), and at the output with the availability of results for consumption - notifications of detected events (4). Internal steps of HN, defined by Activities 3.1 to 3.5, use cloud resources to organize the execution flow. The main event detection methods are present in (3.3), as well as orchestration activities such as configurations (3.1), connection to data *streaming* (3.2), performance monitoring (3.4), and cloud resource monitoring (3.5).

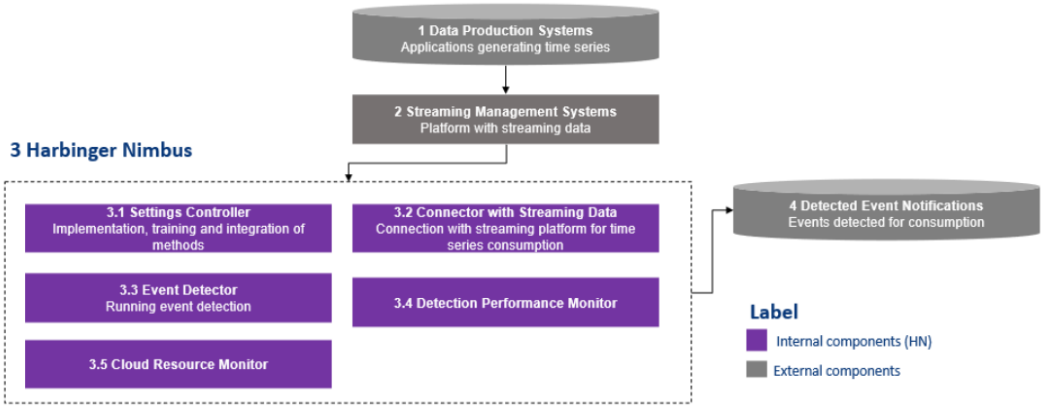

Figure 1. HN: Overview

Once the series is received, through activity (3.2), the connection with the detection methods is carried out. These methods are previously configured through the configuration controller (3.1), which allows the selection and configuration of the parameters of the desired detection method. At the end of the process, the detected events are made available in the cloud for consumption in the form of notifications (activity 4 in Figure 1).

Finally, it should be noted that the proposed architecture is independent of methods or specific platforms for *streaming* and cloud computing. Thus, it is possible to experiment and flexibly compare different methods.

4. Final Remarks

The objective of this work is to explore the *frameworks* gap for cloud integration of event detection methods and not to present a specific method. Implemented detectors in HN can

be compared with the labeled data of the series to calculate the performance. The metrics available are precision, recall, F_1, accuracy, and execution time in seconds.

To compare the versatility of HN, MAD/SR-CNN is available as a service on *Azure* usable via API. At first, this makes its usability simple but limits the options for customization, parameter adjustment, and method combinations. For example, it is impossible to choose the method to be executed, a simple definition of performance metrics, or a combination of methods.

The preliminary implementation opens the door to research and development aimed at evaluating datasets in streaming, varying the time of arrival of the data. HN will also be made available as a service on *Azure*, expanding its potential for use in the context of IoT and digital twins.

In future work, the proposed architecture will be explored more comprehensively to automate the integration of all tasks. Approaches for processing the series in *streaming* will also be analyzed to identify ways to help adapt the methods as the behavior of the series changes over time. Finally, *streaming* platforms can be evaluated to identify possibilities for improving computational cost.

References

Gensler, A. and Sick, B. (2018). Performing event detection in time series with swiftevent: an algorithm with supervised learning of detection criteria. *Pattern Analysis and Applications*, 21(2):543–562.

Guralnik, V. and Srivastava, J. (1999). Event Detection from Time Series Data. In *Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '99, pages 33–42, New York, NY, USA. ACM.

Habeeb, R. A., Nasaruddin, F., Gani, A., Targio Hashem, I., Ahmed, E., and Imran, M. (2019). Real-time big data processing for anomaly detection: A Survey. *International Journal of Information Management*, 45:289–307.

Hiraman, B. R., Viresh, M. C., and Abhijeet, C. K. (2018). A study of apache kafka in big data stream processing. In *2018 International Conference on Information, Communication, Engineering and Technology, ICICET 2018*.

Ogasawara, E., Salles, R., Escobar, L., Baroni, L., Lima, J., and Porto, F. (2021). Online event detection for sensor data. In *Proceedings of the Ibero-Latin-American Congress on Computational Methods in Engineering*.

Ren, H., Xu, B., Wang, Y., Yi, C., Huang, C., Kou, X., Xing, T., Yang, M., Tong, J., and Zhang, Q. (2019). Time-series anomaly detection service at Microsoft. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 3009–3017.

Salles, R., Escobar, L., Baroni, L., Zorrilla, R., Ziviani, A., Kreischer, V., Delicato, F., Pires, P. F., Maia, L., Coutinho, R., Assis, L., and Ogasawara, E. (2020). Harbinger: Um framework para integração e análise de métodos de detecção de eventos em séries temporais. In *Anais do Simpósio Brasileiro de Banco de Dados (SBBD)*, pages 73–84. SBC.

Talagala, P., Hyndman, R., Smith-Miles, K., Kandanaarachchi, S., and Muñoz, M. (2020). Anomaly Detection in Streaming Nonstationary Temporal Data. *Journal of Computational and Graphical Statistics*, 29(1):13–27.

Truong, C., Oudre, L., and Vayatis, N. (2020). Selective review of offline change point detection methods. *Signal Processing*, 167.