Time-series Anomaly Detection Applied to Log-based Diagnostic System Using Unsupervised Machine Learning Approach

Francesco Minarini*✉, Leticia Decker*†✉

*University of Bologna, Bologna, Italy
†INFN (Italian National Institute of Nuclear Physics), Bologna, Italy
francesco.minarini@studio.unibo.it
leticia.deckerde@unibo.it

***Abstract*—Annually, the Large Hadron Collider (LHC) demands a huge amount of computing resources to deal with petabytes of produced data. In the next years, a scheduled LHC upgrade will increase by at least 10 times the computational workload on the Worldwide LHC Computing Grid (WLCG). As a consequence, an upgrade in the computing infrastructure that supports the physics experiments is also required. All IT staff of WLCG computing centers is focused on the development of hardware and machine learning software solutions as log-based predictive maintenance systems. This work presents an original general-purpose diagnostic system to identify critical activity periods of services solving a binary anomaly detection problem using an unsupervised support vector machine approach.**

I. Introduction

Nowadays, it is not possible to split the High-Energy Physics (HEP) development from the computational approaches to data analysis. Yearly, the Large Hadron Collider (LHC) produces dozens of petabytes of data from particle collisions, simulations, metadata, and data analysis. In this way, a huge amount of computing resources are necessary to store exabytes of data, to support a computing throughput of around 10^5 jobs per day followed by an increasing demand for efficient data sharing among computing centers through high-speed networks.

For this reason, the Worldwide LHC Computing Grid (WLCG) was created in order to support the HEP experiments. The network infrastructure is essential support to the LHC discoveries. Nonetheless, the resource requests are deemed to blast in the near future after the *Long Shutdown 2*, a scheduled LHC upgrade that aims to increase the experiment luminosity by a factor of 10 over its designed value, resulting in an increasing of produced data [1].

In HEP scattering theory, luminosity ($\mathcal{L}$) is the ratio of events (dN) detected through the cross-section (σ) over time (dt) [2]

$$\mathcal{L} = \frac{dN}{dt} \tag{1}$$

The increasing of $\mathcal{L}$ generates, at least, a proportional increasing of the data produced by the HEP experiments, and consequently, the workload of the data processing over the Grid. This new scenario implies a hard technical challenge: to keep the infrastructure working from Run-3 and Run-4 stages of the High-Luminosity LHC project (HL-LHC) [1].

The *HEP Software Foundation* (HSF) released a road-map document describing the actions needed to prepare the Grid to support the HL-LHC upgrade [3]. As a result, the *Operational Intelligence group* (OpInt) was created as a predictive maintenance task force to improve the WLCG quality of service (QoS) via analytics and machine learning tools. Its final goal is to build up an automatised log-based event-oriented predictive maintenance system [4].

The scenario of this work is the WLCG tier-1 computing center of the Italian Institute of Nuclear Physics (INFN-CNAF). Here, we propose a solution to the anomaly detection problem applied on INFN-CNAF logs [5], [6]. The objective of this work is to identify critical time-windows of running services though a general-purpose solution.

The rest of this paper is organized as follow. Section II shows the context of this project and the previous activities at INFN-CNAF. Section III describes the service that generated the used logs. Section IV presents the prototype to detect anomalies at the log data. Section V reports the methodology used to generate the result in Section VI. Conclusions and future works are described in Sec VII.

II. HL-LHC Computing

The WLCG is a global collaboration involving more than 170 computing centers over 42 countries linked in a grid infrastructure. Its mission is to provide a common middleware for physics experiment, in which it is possible to share data, and to run applications on computing facilities.

According to HSF, the success of the HL-LHC project depends on the development of support software solutions. They have to be scalable and efficient in order to keep the computational stability in the long term [7].

The INFN-CNAF research group is focused on providing general-purpose predictive maintenance solutions. In [8], an unsupervised machine learning system based on the Elastic Stack Suite was presented. This system is characterized by parsing and cataloguing the log data.

The online anomaly detection is approached in granular fuzzy-set classifiers [9], [10], [11] applied to a 4-class detection problem, and can be used as a pre-processing stage to log parser or text processing solutions.

In this work, we propose a service-based system diagnosis solution through a binary classifier of the system behaviour using an One-class Support Vector Machine approach [7]. To solve the anomaly detection problem [12], [13], a widely discussed topic in predictive maintenance issues, we used the volatility of the logging activity as a metric.

III. StoRM service

StoRM [14] is the storage management service adopted and developed by the INFN-CNAF Tier-1 to support the WLCG initiative. Its aim is to provide high performance to distributed file systems [15], [16]. In particular, it is used by HEP experiments, in which each experiment can customize it according to its own needs.

StoRM is based on the Storage Resource Manager (SRM) protocol that divides operations in synchronous and asynchronous. The first one is typically associated with namespace operations as, e.g. *srmLs* and *tesrmMkdir*. In its turn, the second one is responsible to manage specific functionalities related to the data as transferring, reading, and writing in the distributed file system. Some examples of these operations are *srmPrepareToPut* and *srmPrepareToGet*.

The service is structured in two main stateless components: the Front-End (FE), and the Back-End (BE). They are described in the Sub-section III-A and III-B, respectively. Fig. 1 illustrates the relation between FE and BE modules and storage systems.

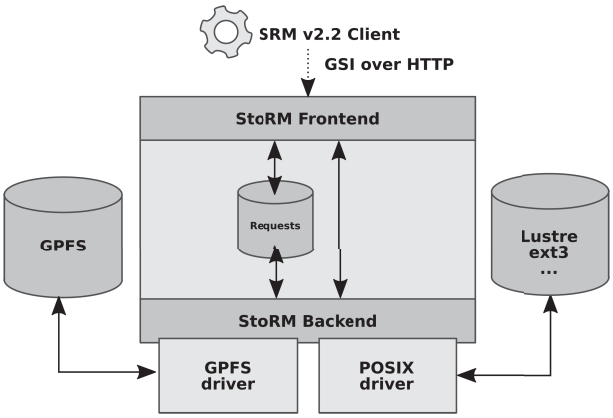

Fig. 1. Abstract of the StoRM components and the relationship between FE and BE components. The client request is taken through the FE interface, who decides, according to its type, to process or forward it to BE module. The memory access is done through the BE component using the access drivers to the distributed file system composed by different technologies [11].

A. StoRM front-end

The FE module provides to the user the access to the SRM web service interface, managing the user credentials, controlling process authentications, storing SRM data request, and retrieving operation status. Whenever there is a new SRM request, FE writes a new log line reporting the occurrence, linking request tokens to BE operations.

An example of such log file is shown in Fig. 2. Each line contains the timestamp, the thread that manages the request, the type of the message (*INFO*, *WARN*, *ERROR*, or *NONE*), the *request-id*, and the actual content of the message.

B. StoRM back-end

Since it manages all synchronous and asynchronous SRM functionalities, allowing interaction with other Grid elements, the BE module can be considered as the main StoRM component. Typical BE log files messages are composed by the requester of the operation (*DN*), the locations of involved files (*SURL*, i.e. the storage *URL* on the WLCG Grid), and the operation status.

Moreover, the structure of the BE log message relies on the StoRM implementation version associated with a physics experiment.

C. Maintenance System

Maintenance systems can be classified in: (1) reactive, (2) preventive, (3) predictive, and (4) advanced maintenance [17]. Reactive maintenance (1) also known as breakdown maintenance, refers to procedures done after failure occurrences in order to restore the system stability, having a palliative impact.

Preventive maintenance (2) refers to procedures performed in order to decrease the likelihood of a system failure. It can involve both periodic equipment overhauls and preventive replacement of devices, and can increase unnecessarily the costs.

Predictive maintenance (3) involves methods that make a prediction of failure occurrences according to system outputs. Related to service monitoring, in general, it is used logs from related system as input. And, finally, the advanced maintenance (4) deals with a combination of the previous approaches.

The maintenance problem is centered in diagnosis and prediction applications. In the first case, the software solution is focused in to identify the actual system status or identify the failure occurrence type. In the second case, the approaches predict the future system status or the occurrence of a failure. The present work is a diagnosis solution, and because of that, it is classified as reactive maintenance.

IV. Prototype of System Maintenance

In this section, we present our solution to anomaly detection problem based on an One-Class Support Vector Machine (OC-SVM) implementation. This method is a binary unsupervised classification used to differ normal and anomalous periods of a running service.

The solution is focused in the classification process. During the pre-processing stage, it is extracted the volatility of the logging activity from the raw log data. After that, the metric sample is classified in normal or anomalous occurrence in the related time period. The process is detailed as follow.

A. Classification Phase

The work is based on the processing of 3 time series. At first, there is the raw log data stream (1) used to extract the logging activity series (2). The third one is the volatility series (3) derived from the second stream.

```
12/01 03:48:11.701 Thread 41 -  INFO [41eec2e2-80a7-4c1b-82b4-d42ff57f0b7e]: Result for request
 'PTP status' is 'SRM_REQUEST_INPROGRESS'
12/01 03:48:11.717 Thread 13 -  INFO [1b3a9db9-1325-467f-9b8b-68347dbb6ad3]: process_request :
Connection from 2001:1470:ff80:12:8e23:c32e:495d:3846
12/01 03:48:11.849 Thread 13 -  INFO [1b3a9db9-1325-467f-9b8b-68347dbb6ad3]: Request 'BOL statu
s' from Client IP='2001:1470:ff80:12:8e23:c32e:495d:3846' Client DN='/DC=ch/DC=cern/OU=Organic
Units/OU=Users/CN=atlact1/CN=555105/CN=Robot: ATLAS aCT 1' # Requested token '17b26868-7752-4e3
0-bcdb-c05d1f72c1b9'
12/01 03:48:11.852 Thread 13 -  INFO [1b3a9db9-1325-467f-9b8b-68347dbb6ad3]: Result for request
 'BOL status' is 'SRM_REQUEST_INPROGRESS'
```

Fig. 2. Example of the content of the StoRM front-end log file. The common feature of different log data is the timestamp that indicates the time of the log message writing.

1) Logging Activity log^t: In this work, anomaly detection is the identification of the anomalies in log stream

$$log^t = \frac{n^t}{w^t} \tag{2}$$

in which n_t is the number of line written in the log file during the time-window w^t with $t = 1, 2, \ldots$, in which $|w^t|$ is invariable.

2) Volatility v: The volatility v is a measure of the behaviour changes of a feature, in this work, the logging activity. It comes from Econophysics domain, being used to detect oscillations of stock markets [18]. Here, it is used to monitor log-based systems. An increasing v indicates the decreasing of the system stability. The v^k is defined by

$$v^k(log_X^k) = \sqrt{\frac{1}{m}\sum_{j=1}^{m}(\mu(log_X^k) - log_j^k)^2}. \tag{3}$$

in which $log_X = [log^1\ log^2\ \ldots\ log^j\ \ldots\ log^m]$ with $|log_X| = m$ is a logging activity set from log_X^k, a time series of logging activity sets with $k = 1, 2, \ldots$. The v^k is a time series of the standard deviation of the log_X^k, bounded by [0, $+\infty$].

In this work, log_j is the j-th logging activity in the log_X set. We applied v to the log_X^k series with $k = 1, 2, \ldots$. The transformed time-series v_k is used as input to the OC-SVM approach.

3) One-Class Support Vector Machine: OC-SVM is an unsupervised outlier detection method, classifying the new data as similar or different to the data-set. OC-SVM implementations create a hyper-planes in a dimensional space used to separate the anomalous data from the normal. In this way, even though it is not a supervised algorithm, OC-SVM considers the frequent data as normal, generating the decision functions called support vectors.

Classifier methods differ in the strategies to create the frontier among the classes. Basically, OC-SVM method separates the data from the origin, maximizing the distance between the origin and this hyper-plane. It gives a binary function that returns $+1$ to the region similar to data-set, and -1, elsewhere. The classifier was implemented using the *svm.OneClassSVM* library of *sklearn*.

B. Prototype Algorithm

Here, we present the algorithm implemented to solve the anomaly detection problem applied to log-based system maintenance. The algorithm is illustrated as a data flow diagram in Fig. 3, in which, the log data is the input and the anomaly analysis is the output.

Offline Time-series Log-based Anomaly Detection

// Indexes Initialization:
$k = t = 1$;
// Sliding time-windows:
$w_{log} = 5$ //$\rightarrow log^t$,
$w_{vol} = 30$ //$\rightarrow v^k(log_X^k)$,
// Classification Phase
WHILE $not\ endOfFile$ DO
  Read sample set $\mathbf{X}^t = \{x^i \mid i = [\underline{w_{log}}\ \overline{w_{log}}]\}$;
  Extract logging activity feature $\overline{log^t}(\mathbf{X}^t)$ $(Eq.(2))$;
  Increment t;
END
WHILE $k < t$ **DO**
  Calculate $v^k(\{log^t | t \in \underline{w_{vol}}\ \overline{w_{vol}}\})$ $(Eq.(3))$;
  Update k;
END
Using OC-SVM, tag v^k as normal or anomalous;
FOR $k = 1$ *to* $|v^k|$ by $step = 1$ **DO**
  IF v^k is anomalous **THEN**
    Insert v^k in Y;
END
Return Y, the set of anomalous periods;

V. Methodology

This section describes the methodology used in the experiments of the proposed solution. The volatility v transformation applied on the log_X^k cannot distinguish between up and down variations, highlighting just the behaviour changing points.

The v^k series highlights critical states of the system expressed as anomalous data in a specific time interval since it is produced from log^t. It captures trend changing, and can be used to identify precursors of events.

A. Data Pre-processing

A log data is a sequence of lines separated in two parts: (1) timestamp, and (2) log message. Timestamp (1) is related to the moment of log message (2) written in a log file. Each log file have a set of possible log messages [11]. The daily data production depends of the event occurrences during this period, and can reach about millions of lines per day for a single log

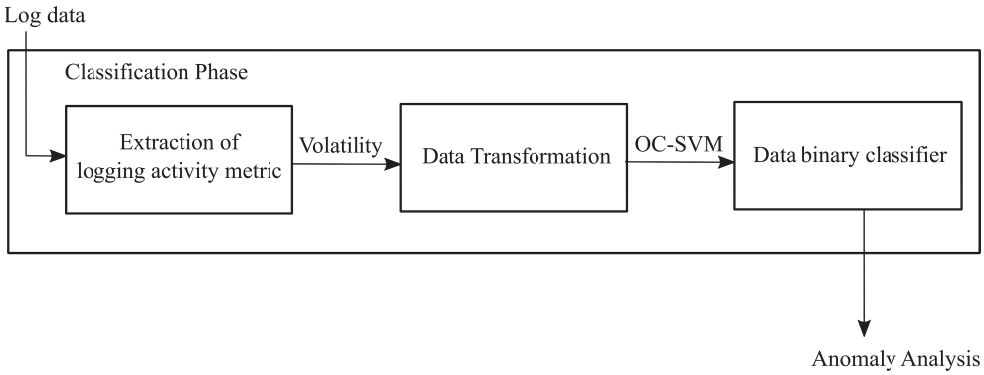

Fig. 3. Data flow diagram of the proposed diagnostic system, highlighting the main steps of the solution, from the raw log data input to the anomaly classification as output.

file of a given service. For this reason, the log data used in this approach is pre-processed using the following operations.

1) Data re-sampling: The time-window w_{log} used to extract the log is 5 minutes (see Table.I).

2) Data scaling: The v^k series are normalized using the highest value of each day (Eq.(3)). This step avoids a scale-factor bias towards higher volatility values.

B. Data-set

There is no specific reason for choose StoRM Service, the front-end log file, or the ATLAS experiment [9]. Any event-drive log data can be used as input for this method as, e.g., the log example in Fig. 2.

The input data were collected from December 2^{nd} to December 8^{th}, 2018. This time interval contains a critical period of StoRM running highlighted by the notification ticket seen in Fig. 5) at the INFN-CNAF support system. The Table I shows the abstract of the log data details.

TABLE I. LOG DATA USED AS INPUT TO DIAGNOSTIC SYSTEM.

ID	Interval of Days	Month	Year	Classification
1	02-04	12	2018	Normal
2	05-08	12	2018	Anomalous

VI. RESULTS

This section describes the results achieved by the simulations of the diagnostic system with the parameters of Table II using the input data described in Table I, including sliding time-windows(tw) of the classification phase.

TABLE II. PARAMETERS USED IN THE EXPERIMENTS OF THE DIAGNOSTIC SYSTEM.

Module	Parameter	Description	Value
Feature Extr.	w_{log}	tw*	5 min.
Data Transf.	w_{vol}	Sliding tw	30 min.
Data Transf.	$step_{w_{vol}}$	Step of w_{vol}	30 min.
OCSVM	$kernel$	Parameter	RBF
OCSVM	nu	Parameter	0.2
OCSVM	$gamma$	Parameter	auto

*time-window

The input data 1 is considered normal, and the data 2 is identified as anomalous period (see the reporting ticket in the Fig. 5). The classifications are centered in days, and classification of different days are independent of each others. In this way, it is possible to classify both data-set in micro-periods of regularity and anomalies of a running service.

Fig. 4 shows v^k in different days of the period of interest, highlighting the percentile compositions of each one. As we can see, there is a peak of data variability during the anomalous period (see Table I). Fig. 6 shows log^t vs. $time$, respectively, the 2^{nd}(normal), 6^{th}, 7^{th}, and 8^{th} (anomalous) December. It is possible to see the difference of the patterns of logging activities in both cases (normal and anomalous). In the anomalous period, there are more intense peaks of logging activity interspersed with flat periods, explaining the big variability of the volatility in the Fig. 4, and the bigger range of the value of logging activity in the Fig. 6. On 8^{th} December, the end of anomalous period, we can see the pattern changing in order to approach to the 2^{nd} December model.

Conversely, in the normal periods, it is not possible to see the flat periods. In this sense, logging activities pulse in peaks of anomalies that have a higher probability of to contain useful information to the diagnostic systems.

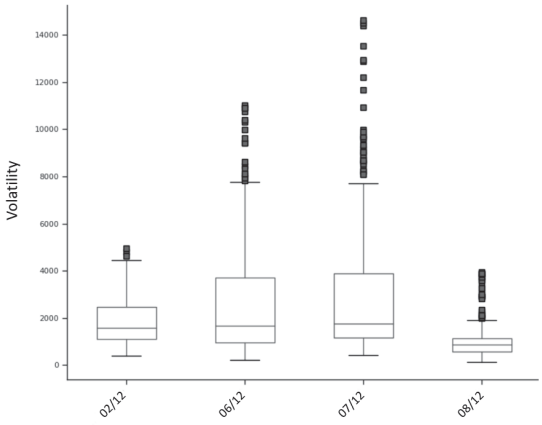

Fig. 4. The variability the volatility of the logging activity (lines per w_{log}) of StoRM Front-End logs over time.

The OC-SVM classifier was implemented using the

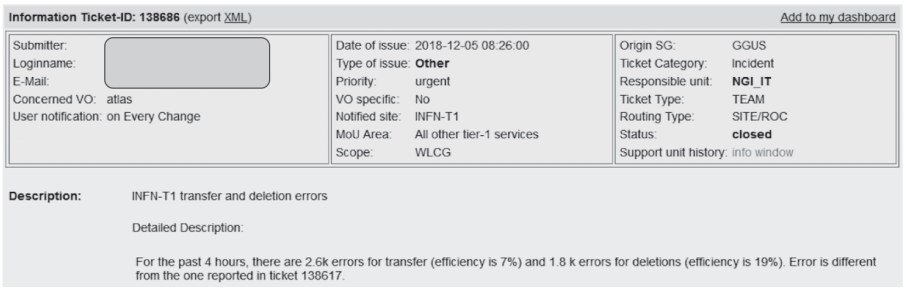

Fig. 5. The ticket generated by users to report the beginning of the anomalous period of the StoRM service.

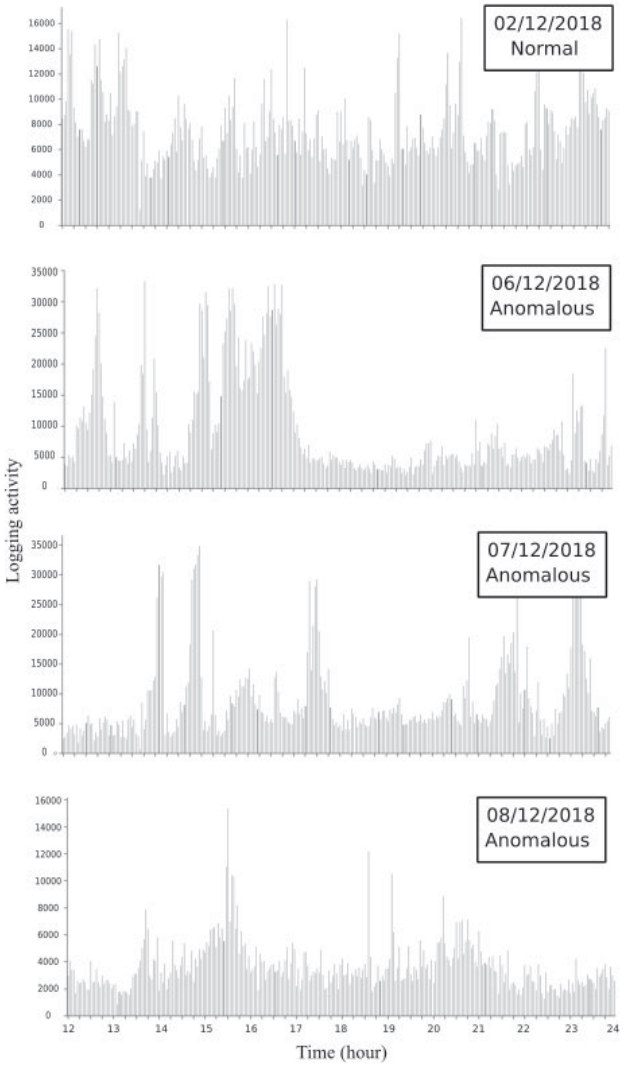

Fig. 6. Logging activity (lines per w_{log}) of StoRM Front-End in a normal and anomalous days.

$sklearn.svm.OneClassSVM$ library (see Table II for details), using 2^{nd} and 7^{th} December data. Fig. 7 shows the v over time for: (1) normal and (2) anomalous scenarios, respectively, top and bottom graphics. The same peaks-flats patterns in (2) can be observed in this figure. This plot highlights the anomalous (square) and normal (circle) periods. It is easy to see a smooth effect of v, separating the data trends of the stochastic fluctuations, when we compare the results in Fig. 4 and 7, facilitating the identification of periods of interest to the diagnostic system (2).

The analysis of the Fig. 4 and 7 are complementary: the y-axis range of the first one can give the identification of the anomalous days that have to be processed in order to extract the anomalous micro-periods. These periods are inputs to a text processing method to extract useful information to a decision making procedure, and that is out of the scope of this paper.

Also in Fig. 7, there are almost-zero v^k intervals. This pattern is classified as anomaly and is related to the frozen service problem, e.g. a period of non-response of a service.

VII. Conclusion

The proposed anomaly detection approach, based on OC-SVM, is able to identify the anomalous period in a normal or anomalous data-set. It is a general-purpose 2-class log-based classifier, having as output time-intervals classified as anomalous in the context of the given data. In the system maintenance context, anomalous periods are related to non-desirable event occurrences in running systems. Data analysis using text processing are future works. A text processing approach possibility is term frequency–inverse document frequency (TFIDF), a naive approach used to extract words (or combination of words) according with their importance in the data-set.

Acknowledgement

The authors would like to acknowledge INFN Bologna for their financial support to this activity.

References

[1] A. Di Girolamo *et al.*, "Operational intelligence for distributed computing systems for exascale science," in *Int Conf on Computing in High Energy and Nuclear Physics (CHEP), AU*, p. 8p, 2020.

[2] W. Herr and B. Muratori, "Concept of luminosity," in *CAS - CERN Accelerator School: Intermediate Course on Accelerator Physics*, 2006.

[3] J. Albrecht et al., "A Roadmap for HEP Software and Computing R&D for the 2020s," *Comput Softw Big Sci*, vol. 3, 2019.

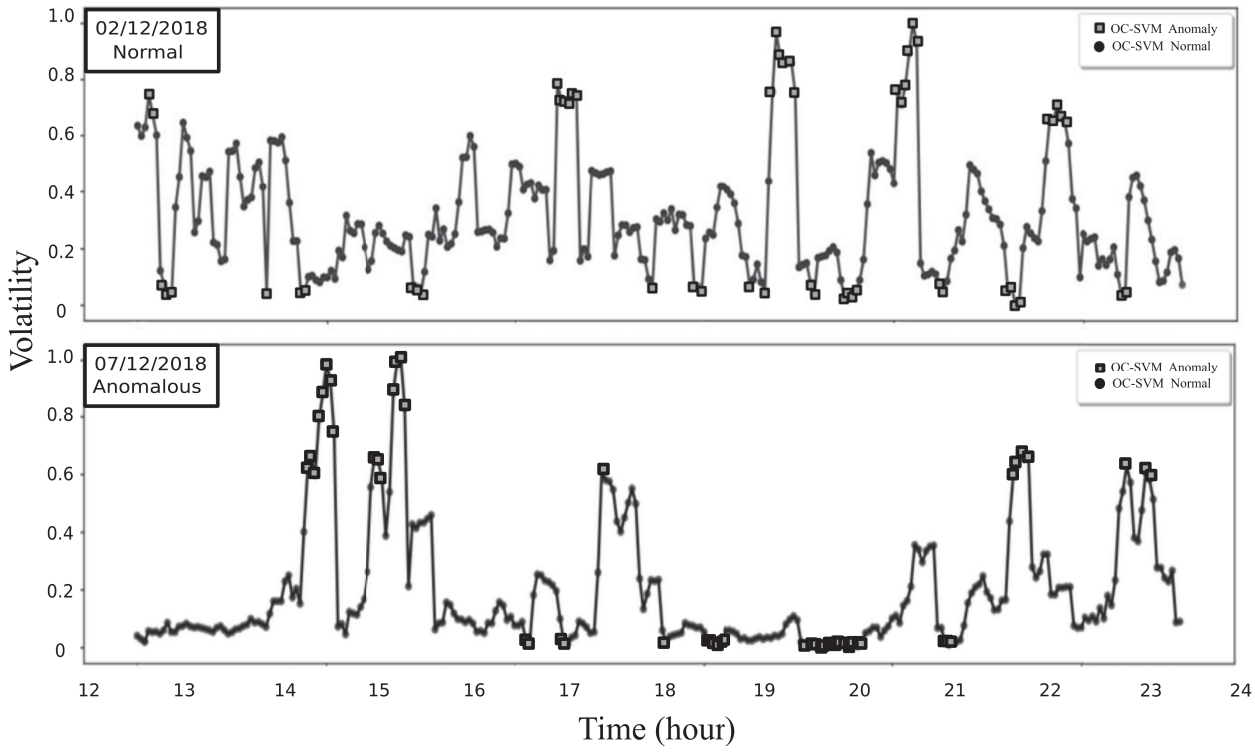

Fig. 7. Volatility (lines per w_{log}) over time of the normal (top) and anomalous (bottom) periods.

[4] L. D. Sousa *et al.*, "Big data analysis for predictive maintenance at the INFN-CNAF data center using machine learning approaches," in *Conf. of Open Innovations Association (FRUCT), Helsinki*, pp. 448–451, 2019.

[5] L. D. Sousa *et al.*, "Event detection framework for wireless sensor networks considering data anomaly," in *IEEE Symp. Comput. and Commun. (ISCC)*, pp. 500–507, 2012.

[6] L. D. de Sousa, "Detecção de eventos em redes de sensores sem fio," Master's thesis, Universidade Federal de Minas Gerais, 2011.

[7] F. Minarini, "Anomaly detection prototype for log-based predictive maintenance at INFN-CNAF," Master's thesis, University of Bologna, 2019.

[8] T. Diotalevi et al., "Collection and harmonization of system logs and prototypal analytics services with the elastic (ELK) suite at the INFN-CNAF computing centre," in *International Symposium on Grids & Clouds (ISGC). Taipei, Taiwan: Proceedings of Science*, 2019.

[9] L. Decker, D. Leite, L. Giommi, and D. Bonacorsi, "Real-time anomaly detection in data centers for log-based predictive maintenance using an evolving fuzzy-rule-based approach," in *IEEE World Congress on Comput. Intell. (WCCI, FUZZ-IEEE), Glasgow*, p. 8p, 2020.

[10] D. Leite, L. Decker, M. Santana, and P. Souza, "EGFC: Evolving Gaussian fuzzy classifier from never-ending semi-supervised data streams - with application to power quality disturbance detection and classification," in *IEEE World Congress on Computational Intelligence (WCCI – FUZZ-IEEE), Glasgow*, p. 8p, 2020.

[11] L. Decker, D. Leite, F. Viola, and D. Bonacorsi, "Comparison of evolving granular classifiers applied to anomaly detection for predictive maintenance in computing centers," in *IEEE Conference on Evolving and Adaptive Intelligent Systems (EAIS), Bari*, p. 8p, 2020.

[12] D. Leite, "Comparison of genetic and incremental learning methods for neural network-based electrical machine fault detection," in *Predictive Maintenance in Dynamic Systems* (E. Lughofer and M.Sayed-Mouchaweh, eds.), pp. 231–268, Cham: Springer, 2019.

[13] S. Silva, P. Costa, M. Gouvea, A. Lacerda, F. Alves, and D. Leite, "High impedance fault detection in power distribution systems using wavelet transform and evolving neural network," *Electric Power Systems Research*, vol. 154, pp. 474 – 483, 2018.

[14] *The StoRM project*. https://italiangrid.github.io/storm/index.html.

[15] A. Carbone, L. dell'Agnello, A. Forti, A. Ghiselli, E. Lanciotti, L. Magnoni, M. Mazzucato, R. Santinelli, V. Sapunenko, V. Vagnoni, and R. Zappi, "Performance studies of the storm storage resource manager," in *Third IEEE International Conference on e-Science and Grid Computing (e-Science 2007)*, pp. 423–430, 2007.

[16] L. Magnoni, R. Zappi, and A. Ghiselli, "Storm: A flexible solution for storage resource manager in grid," in *2008 IEEE Nuclear Science Symposium Conference Record*, pp. 1971–1978, 2008.

[17] F. Trojan and R. Marçal, "Proposal of maintenance-types classification to clarify maintenance concepts in production and operations management," *Journal of Business Economics*, vol. 8, pp. 562–574, 2017.

[18] S. Micciche', G. Bonanno, F. Lillo, and R. Mantegna, "Volatility in financial markets: Stochastic models and empirical results," papers, arXiv.org, 2002.