EPJ Web of Conferences **245**, 07016 (2020) https://doi.org/10.1051/epjconf/202024507016
CHEP 2019

Anomaly detection using Unsupervised Machine Learning for Grid computing site operation

Tomoe Kishimoto[1*], *Junichi* Tnaka[1], *Tetsuro* Mashimo[1], *Ryu* Sawada[1], *Koji* Terashi[1], *Michiru* Kaneda[1], *Masahiko* Saito[1], and *Nagataka* Matsui[1]

[1]International Center for Elementary Particle Physics, The University of Tokyo, Japan

Abstract. A Grid computing site is composed of various services including Grid middleware, such as Computing Element and Storage Element. Text logs produced by the services provide useful information for understanding the status of the services. However, it is a time-consuming task for site administrators to monitor and analyze the service logs every day. Therefore, a support framework has been developed to ease the site administrator's work. The framework detects anomaly logs using Machine Learning techniques and alerts site administrators. The framework has been examined using real service logs at the Tokyo Tier2 site, which is one of the Worldwide LHC Computing Grid sites. In this paper, a method of the anomaly detection in the framework and its performances at the Tokyo Tier2 site are reported.

1 Introduction

A Grid computing site is composed of various services including Grid middleware, such as Computing Element (CE) and Storage Element (SE). The Grid middleware is a software stack, which allows remote users to access distributed computing resources in the Grid. Therefore, site administrators are required to ensure stable operations of the services to utilize the computing resources efficiently. Text logs produced by the services provide useful information for understanding the status of the services. However, it is a time-consuming task for the site administrator to monitor and analyze the service logs every day. Therefore, a support framework has been developed to ease the site administrator's work. The framework is designed to detect anomaly logs, which could indicate a problem of the service, using Machine Learning (ML) techniques and alert the site administrators.

To our knowledge, there are several reports on the anomaly detection using ML techniques for Grid computing site operation in the high energy physics community. An approach focused on time series anomaly detection using an embedded ML tool in the Elastic Stack Suite [1] is reported in Ref. [2]. Another approach using supervised ML to predict anomalies in text logs is reported in Ref. [3]. In this paper, we have introduced a word embedding technique and a clustering algorithm, which are both unsupervised ML, to detect anomaly logs. Unsupervised ML techniques are selected because anomaly logs are not known in advance as they evolve over time, e.g new security incidents, and therefore a ML model cannot be trained based on pre-labeled data. The framework has been examined using real service logs at a Grid computing site in the University of Tokyo, which is one of the Worldwide LHC

*e-mail: tomoe@icepp.s.u-tokyo.ac.jp

Computing Grid (WLCG) sites. In this paper, a method of the detection of anomaly logs in the framework and its performances at the Grid computing site are reported.

The paper is organized as follows, Section 2 describes the system configuration of the Grid computing site in the University of Tokyo. Section 3 gives an overview of the framework and ML algorithms. Section 4 shows a test of the ML algorithms using sshd and cron logs. Section 5 shows performances of the anomaly detection using real service logs of the Grid computing site. Section 6 gives the conclusion and discussion.

2 System configuration of the Tokyo Tier2 site

The International Center for Elementary Particle Physics (ICEPP) at the University of Tokyo operates a computing center for the WLCG. The computing center supports the ATLAS experiment as a Tier2 site (hereafter, the computing center is called Tokyo Tier2 site). Hardware devices in the Tokyo Tier2 site were upgraded every three years to satisfy the requirements of the ATLAS experiment. In 2019–2021, 7680 CPU cores and 10.56 PB disk storages are reserved for the ATLAS experiment.

For the Grid middleware in the Tokyo Tier2 site, the ARC-CE [4] is deployed as the CE in front of the HTCondor [5] batch job scheduler. The disk storage system is managed by the DPM [6] as the SE. The DPM supports relevant protocols (e.g. GridFTP) for file management and access. The Tokyo Tier2 site is one of the biggest DPM sites in the WLCG. Real text logs produced by the DPM at the Tokyo Tier2 site are used to evaluate performances of the anomaly detection of the framework in the following sections.

3 Overview of the anomaly detection framework

Figure 1 shows an example of text logs produced by the DPM. The logs show a typical format with time stamp, severity and log message. In this example, it can be confirmed that there was a problem with authorization at time May 29 14:05:51 (log message is colored in red). Thus, text logs provide useful information to understand the status of the service and solve the problem. This log should be detected as an anomaly because the log was generated due to a misconfiguration of the DPM and the service was degraded.

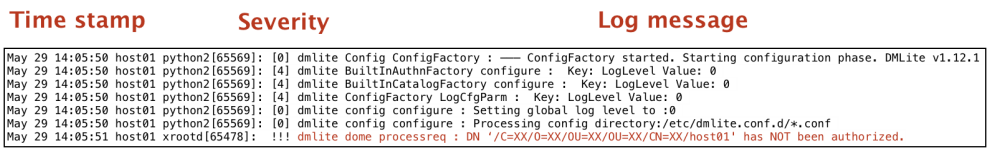

Figure 1: An example of DPM logs.

A log analysis using scripts is a simple approach to detect problems but this approach does not scale for complicated logs. For example, a large amount of error messages, which is more than one million, are observed at the Tokyo Tier2 site every day. It is difficult to set criteria for all possible error logs to detect serious problems, which need to be investigated. Therefore, a support framework using ML techniques based on log similarity has been developed to detect anomaly logs automatically. The framework consists of several components as shown in Figure 2. The following subsections describe each component.

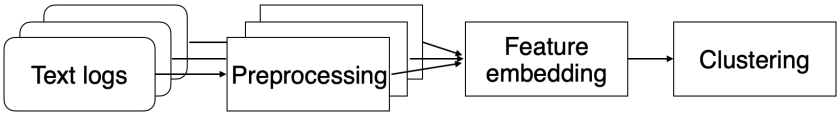

Figure 2: Components of the anomaly detection framework.

Table 1: An example of output by Logwatch for the DPM logs.

Error counts	Error messages
336780 time(s)	globus-gridftp-server : dmlite DmStatus : Info: [#00.000002] DPM_LFN not found
333902 time(s)	xrootd : dmlite DmStatus : Info: [#00.000002] replica DPM_LFN not found
333900 time(s)	globus-gridftp-server : dmlite DmException : DmException(..):[#00.000002] Error when issuing request to 'http://host01:1094/domehead/command/dome_getstatinfo'. Status 404. DavixError: 'HTTP NUMBER : File not found '. Response (NUMBER bytes): 'File not found on rfn: DPM_LFN err: NUMBER what: '[#00.000002] replica DPM_LFN not found'
...	...

3.1 Preprocessing

The service logs on each server are parsed and aggregated using Logwatch [7] software every day. In Logwatch, dedicated scripts are required for each service to parse and aggregate logs. Template scripts are available by default for major services such as sshd, cron and httpd. A dedicated script for the DPM logs has been developed for this study. The script selects only error messages and replaces meaningless words with common words so that the aggregation works correctly. For example, the following replacements are performed:

- file name and path (e.g. /data00/201905015/file_name) → DPM_LFN,
- memory address (e.g. 0xabdef) → NUMBER.

Table 1 shows an example of output by Logwatch for the DPM logs. The output shows counts for each error log message after the word replacements. The information of error counts is also used in subsequent ML processes to be sensitive to unusual error counts. About 50 types of error messages are observed every day for the DPM at the Tokyo Tier2 site. The outputs by Logwatch are collected from each server and saved into a sqlite3 database.

3.2 Feature embedding

The text logs are converted to feature vectors using a word embedding technique. Doc2Vec [8] algorithm in gensim [9], which is an unsupervised ML algorithm, that is used in the framework. Doc2Vec algorithm is an adaption of Word2Vec [10] algorithm. The algorithms generate feature vectors from documents or words using a neural network that predicts a target word from context. In this algorithm, similar text logs should be mapped close to each other in feature vector space. Therefore, anomaly logs are expected to be mapped away from normal logs because anomaly logs are assumed to contain different words than normal logs

in this study. Based on past operational experiences of the site, this assumption that anomaly logs contain different words is considered to work well for the DPM logs. The behavior of log similarity is tested using simple logs as discussed in Section 4. The following hyper parameters of the algorithm are tuned in this study:

- **vector_size**: dimensionality of the feature vectors (300),
- **window**: the maximum distance between the current and predicted word (7),
- **epochs**: the number of iterations (epochs) over the corpus (500).

The numbers in brackets are selected values after the hyper parameter tuning described in Section 5. The default values are used for the other hyper parameters in gensim.

3.3 Clustering

The feature vectors generated by Doc2Vec algorithm are identified as normal events or anomaly events using a clustering algorithm. IsolationForest [11] algorithm in scikit-learn [12] is used in the framework. In this algorithm, decision trees are defined with random features and values. The algorithm assumes that anomaly events tend to result in small number of paths in the decision tree. The number of clusters is not needed to be defined in advance as an input parameter of the algorithm. In the framework, the events identified within clusters by the algorithm are considered normal events and the events identified as outliers are considered anomaly events. The following hyper parameters are tuned in this study:

- **n_estimators**: the number of base estimators in the ensemble (10),
- **contamination**: the amount of contamination of anomaly events (0.11).

The numbers in brackets are selected values after the hyper parameter tuning described in Section 5. The default values are used for the other hyper parameters in scikit-learn.

4 Test of the ML algorithms using sshd and cron logs

The feature embedding and clustering are key components in the framework. These ML algorithms are tested using simple sshd and cron logs to confirm their behavior. Table 2 shows dummy sshd logs for the test. The logs consist of messages + numbers, e.g. “Users logging in through sshd: root: xxx:xxx:xxx:1:" + “288 Time(s)". This is considered as a good test sample because the DPM logs in the Table 1 also consist of messages + numbers. The logs labeled as Anomaly1, Anomaly2 and Anomaly3 are considered as anomaly logs in this test. The differences between normal and anomaly logs are indicated in bold. The sshd logs labeled as Anomaly1 and Anomaly2 show larger login counts compared to the normal logs and the sshd log labeled as Anomaly3 shows another login counts from an unusual host. The dummy cron logs are also defined under the same context as a second dataset.

In some cases, numbers in logs are important information for the anomaly detection. For example,“1000 times login" message (Anomaly2) is very suspicious and need to be detected as anomaly. It is also considered that “1000 times login" message is more suspicious compared to “300 times login" message when the normal message is “288 times login". To introduce a similarity between numbers, numbers are converted to other words in this study. The conversion is performed so that close numbers show common words as follows.

- “25" → “**digit2to2 bit2to2 digit2to1** bit2to2to5",
- “26" → “**digit2to2 bit2to2 digit2to1** bit2to2to6".

Table 2: Dummy sshd logs to test the ML algorithms.

Label	Text logs	# of samples
Normal	Users logging in through sshd: root: xxx:xxx:xxx:1 : 288 times	91
Anomaly1	Users logging in through sshd: root: xxx:xxx:xxx:1 : **300 times**	3
Anomaly2	Users logging in through sshd: root: xxx:xxx:xxx:1 : **1000 times**	3
Anomaly3	Users logging in through sshd: root: xxx:xxx:xxx:1 : 288 times, **root: xxx:xxx:xxx:2 : 10 times**	3

The code of the number conversion is available in Ref. [13].

The defined sshd and cron logs are converted to two-dimension vectors using Doc2Vec algorithm and clustered using IsolationForest algorithm in the framework. The sshd and cron logs are processed separately as different set of samples.

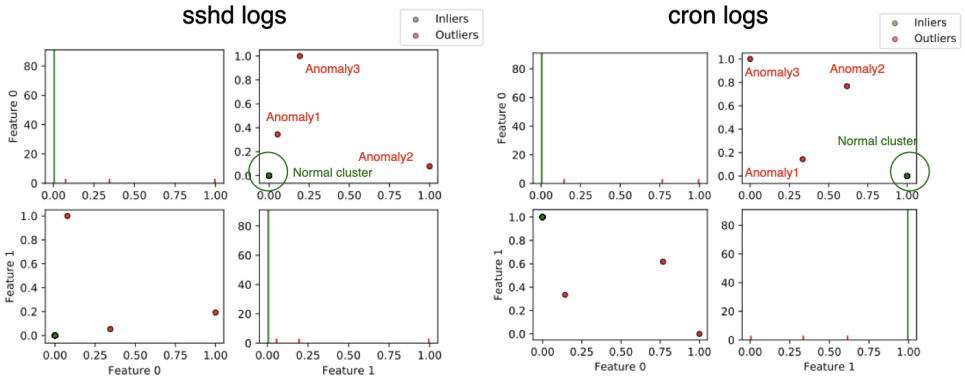

Figure 3: The distribution of two-dimensional word vectors for the sshd (left) and cron (right) logs.

Figure 3 shows the distribution of two-dimensional feature vectors for the sshd and cron logs. The green (red) points show samples identified as inliers (outliers) by the IsolationForest algorithm. In the framework, the samples identified as inliers are considered normal logs and the samples identified as outliers are considered anomaly logs. For both sshd and cron logs, the logs predicted to be anomaly matches the logs pre-labeled as Anomaly1, Anomaly2 and Anomaly3. Therefore, the following points are confirmed by this test:

- Logs containing different words than the normal logs are mapped away from the normal log cluster by the Doc2Vec algorithm as expected,
- IsolationForest algorithm detects outliers based on the feature vectors, then the outliers can be considered anomaly logs.

The number conversion was performed to introduce a number similarity, but its impact for the word embedding is not clear in this simple test. Well-designed logs need to be defined to validate the similarity. It should be a future study.

Table 3: The definition of data for performance studies using the DPM logs.

Date	# of samples	# of anomalies	Data type
2019 May 29 – 2019 Aug 7	71	7	tuning data
2019 Aug 8 – 2019 Aug 31	24	2	validation data

Table 4: The detected counts in confusion matrix for the tuning data.

		Pre-defined label	
		Anomaly	Normal
Predicted label	Anomaly	true positive: 6	false positive: 2
	Normal	false negative: 1	true negative: 62

5 Performances for DPM logs

The framework has been examined using the real DPM logs at the Tokyo Tier2 site. The DPM logs are aggregated using Logwatch every day as described in Section 3.1. The error counts in Table 1 are converted to other words based on the rule, which is discussed in Section 4, to detect unusual error counts. All aggregated logs in a day are considered as an input sample of the ML algorithms, i.e. 1 day = 1 sample in this study. Table 3 shows the definition of samples for the performance studies.

The hyper parameters of Doc2Vec and IsolationForest algorithms are optimized using the tuning data. There were 7 days operational issues in this tuning data period at the Tokyo Tier2 site. For example, a configuration of disk quota was wrong, then many data transfers failed during 2019 Aug 2–3. The data samples in these 7 days are labeled as anomaly data and the hyper parameters are tuned to maximize the F1 score using the tuning data. The F1 score is defined as follows to measure an accuracy of prediction:

$$\text{F1 score} = \frac{2 \cdot \text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} \tag{1}$$

$$\text{precision} = \frac{\text{true positive}}{\text{true positive} + \text{false positive}}, \ \text{recall} = \frac{\text{true positive}}{\text{true positive} + \text{false negative}} \tag{2}$$

The F1 score reaches its best value at 1 and worst score at 0. Table 4 shows the detected counts in confusion matrix for the tuning data after the hyper parameter tuning. The observed F1 score for the tuning data is 0.8.

Then, the framework with the best hyper parameters is applied to the validation data to predict anomaly samples. The target date, which is the date we want to know if anomaly or not, is defined in this evaluation as shown Figure 4. If we consider an actual operation situation, the only samples before the target date is available for the ML training. Thus, the sample for 70 days before the target date is used with the best hyper parameters to train the ML model, and then the ML model predicts a result of the target date.

By this procedure, the framework detects 4 data samples, which correspond to Aug 24, 27, 28, 29, as anomaly data. The two data samples for Aug 27, 28 are pre-labeled as anomaly because the DPM at the Tokyo Tier2 site was upgraded on Aug 27, then the DPM showed new type of error logs after the upgrade. The anomaly detection of the sample for Aug 29 is also reasonable because the new type of error logs were still detected. The days after Aug 29 are not detected as anomaly, while the new error logs were still detected. This result can

be explained by the fact that the clustering algorithm learns the new error logs as normal behavior. An explanation of the anomaly detection for Aug 24 sample is unclear at this time and it is a future subject. The observed F1 score with the initial labeling is 0.67. If the sample for Aug 29 is considered as a true positive because the anomaly detection of the sample is reasonable, the framework shows a better performance with the F1 score = 0.86.

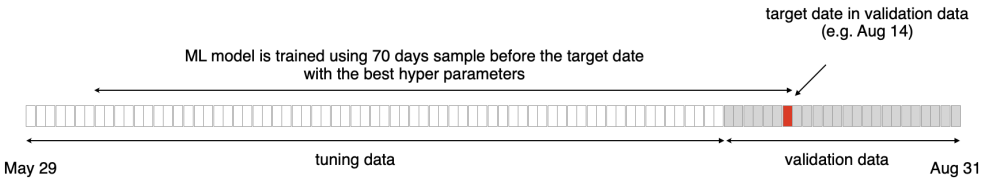

Figure 4: Illustration of the target date and training samples.

6 Conclusion and discussion

Anomaly detection using unsupervised ML techniques for Grid site operations is reported. Doc2Vec and IsolationForest algorithms are used in the framework to detect anomaly logs. The framework was tested using simple sshd and cron logs to confirm the behavior of ML algorithms. The framework was also applied to the real DPM logs at the Tokyo Tier2 site. A reasonable F1 score, which is 0.67, is observed in the validation data.

The data labeling was still required to optimize hyper parameters even if unsupervised ML is used as a concept, and more data samples are needed to understand the ML behavior and improve the method. For the future studies, system information such as CPU usage, memory usage and network traffic can be included to perform more precise anomaly detection.

References

[1] ELK stack: *https://www.elastic.co/jp/what-is/elk-stack*

[2] T. Diotalevi et al., Proceedings of International Symposium on Grids & Clouds 2019, PoS(ISGC2019)027

[3] L. Giommi et al., Proceedings of International Symposium on Grids & Clouds 2019, PoS(ISGC2019)003

[4] M.Ellert et al., Future Generation Computer Systems **23**, 219-240 (2007)

[5] Douglas Thain and Todd Tannenbaum and Miron Livny, Concurrency - Practice and Experience **17** 2-4, 323-356 (2005)

[6] A. Manzi et al., J. Phys.: Conf. Ser. **898**, 062011 (2017)

[7] Logwatch: *https://sourceforge.net/projects/logwatch/*

[8] Quoc V Le and Tomas Mikolov, In ICML **14**, 1188-1196 (2014)

[9] Radim Řehůřek and Petr Sojka, Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks, 45-50 (2010)

[10] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, Jeff Dean, Advances in Neural Information Processing Systems 26, 3111–3119 (2013)

[11] Liu, Fei Tony and Ting, Kai and Zhou, Zhi-Hua, Proceedings of the 2008 Eighth IEEE International Conference on Data Mining, 413–422 (2009)

[12] Pedregosa et al., Journal of Machine Learning Research **12**, 2825–2830 (2011)

[13] Github link: *https://github.com/tkishimoto/numsimword*