IOP Conference Series:
Materials Science and
Engineering

PURPOSE-LED
PUBLISHING™

PAPER • OPEN ACCESS

Detection of network anomalies in log files using machine learning methods

To cite this article: V A Skazin *et al* 2021 *IOP Conf. Ser.: Mater. Sci. Eng.* **1069** 012021

Detection of network anomalies in log files using machine learning methods

V A Skazin, A V Pavlychev, S S Zotov

Polytechnic Institute, Far Eastern Federal University, Vladivostok, 10 Ajax Bay, Russky Island, Russia

E-mail skazin_va@dvfu.ru

Abstract. Detection of network anomalies plays an important role in ensuring information security and countering unauthorized access to information infrastructure, including critical facilities. Detecting of abnormal events in log files is complicated by the fact that individual events without any context may be uninformative. The growing importance of log file analysis in large computer systems requires the development of automated methods for processing unstructured data that retrieves information from large log files without human intervention. This article discusses K-means data clustering method and Isolation forest and OSVM machine learning algorithms in terms of searching network anomalies in network log files in order to detect malicious domains.

Keywords: Machine learning, Isolation Forest, One class SVM, K-means, Text data processing, GrayLog, Network anomaly search

1. Introduction

Nowadays, there is a great interest in detection of "abnormal" states in data sets using machine learning methods. The process of detecting "abnormal" states is known as anomaly or outlier detection. The initial definition of this process was given by Grubbs in 1969 [1]: “an outlying observation, or outlier, is one that appears to deviate markedly from other members of the sample in which it occurs”. Although this definition is still relevant today, the goals for detecting outliers are different now. Initially, the main task of detecting anomalies was to remove outliers from the training data, since the pattern recognition algorithms were quite sensitive to them. This procedure is also called data cleansing. After the development of more reliable classifiers, attention to detecting anomalies has diminished significantly. However, critical point was around 2000, when researchers paid more attention to the anomalies themselves, as they are often associated with specific events or suspicious data records. Since then, new algorithms for detecting anomalies have been actively studied. Some algorithms will be studied in this article. Since the purpose of using anomaly detection algorithms has changed, the Grubbs’ definition can be expanded by two important characteristics:

- Anomalies differ from the norm in terms of their characteristics
- Anomalies are rare in the dataset compared to normal instances.

Currently, anomaly detection algorithms are widely used in different areas and often improve traditional rule-based detection systems.

The prime example of using an anomaly detection algorithm is modern intrusion detection tools. A first stage presents tracking of network traffic and server application activity by pattern matching according to pre-set rules (heuristic analysis). In the second stage, the anomaly detection module attempts to identify unknown suspicious activity (behavioral analysis), which identifies potential intrusion attempts and exploits.

Anomaly detection algorithms cover different requirements. In some cases, the simple algorithms are used, which must be very fast to work in near-real-time mode. In other cases, complex algorithms are used when detection efficiency is important due to the high criticality of anomaly missing. Intrusion detection tools may be classified in relation to the time when the anomaly should be detected. In addition to post-incident analysis and real-time detection, there is predictive motivation, known as early warning. Obviously, this is the most difficult task of detecting anomalies. However, major incidents are often preceded by minor signs that can be detected.

In this article, we present a comparative assessment of some algorithms for detecting anomalies. We describe the strengths and weaknesses of the algorithms in terms of their usefulness for specific application scenarios. As a result, this work is aimed to choose a suitable algorithm for detecting uncontrolled anomalies for the intrusion detection problem.

2. Methodology for finding anomalies in system logs

The algorithm for finding anomalies using log file analysis consists of four main stages: collecting of log files, pre-processing data, clustering data and searching for anomalies.

Collecting of log files. Security systems in modern corporate networks generate logs with a large amount of data about the state of the network and its processes. The first stage includes selection of the system for collecting log files and the main parameters which will be included in the final upload. Data can be uploaded to a csv file for more convenient analysis.

Preliminary data processing. As a rule, uploaded CSV files of system logs contain poorly structured data and are often presented in an incorrect format. The purpose of preprocessing is to remove events or features that are not used in the algorithm in the future, as well as to present the necessary data of the algorithm to a unified form for better clustering in the future.

Clustering. Clustering is the task of grouping a set of objects in **clusters.** Each cluster must have "similar" objects, and objects must be different from each other. The most popular clustering algorithm used in anomaly search methods is K-means.

K-means. This clustering algorithm divides data into appropriate groups based on the information available in the algorithm. Data is divided into K remote clusters in order to produce effective data mining results. Each cluster has a center called a centroid. The data point is grouped into a specific cluster based on how close the objects are to the centroid.

The K-means algorithm iteratively minimizes the distance between each data point and its centroid to find the most optimal solution for all data points. The stages of the algorithm are:

1. K-random points of the data set are selected as centroids;
2. Distances between each data point and the centroids K are calculated and stored;
3. Each point is assigned to the nearest cluster according to the calculation of the distance;
4. New positions of the cluster centroids are updated similar to finding the average value in locations;
5. If the location of the centroids has changed, the process is repeated from step 2 until the calculated new center remains the same, which indicates that the cluster members and centroids are now installed.

Minimum distances between all points are found if the data points were separated in such way as to form the most compact clusters with the least variance. In other words, no other iteration can have a smaller average distance between the centroids and the data points inside them.

The defined K-means algorithm aims to minimize the objective function. In this case it is presented as the squared error function:

$$\boldsymbol{J = \sum_{i=1}^{k} \sum_{j=1}^{n} (||x_i - u_j||)^2 = 1}$$

where, $||x_i - u_{j||}$ is the Euclidean distance between the point x_i and the centroid u_j, repeated over all points *k* in the *i* cluster for all *n* clusters.

Basically, the objective function tries to select centroids which minimize the distance to all points in its corresponding cluster, that increases accuracy of centroids for the surrounding cluster of data points. This number is set based on the results of previous research and theoretical considerations.

Search for anomalies. The results of clustering the initial data set can be used in machine learning methods to search for anomalous data. This paper discusses the following methods: Isolation Forest, OSVM.

Isolation forest. This method builds a set of trees. Each branch corresponds to a chosen random attribute and an outlier. The tree is built until each object (or the specified number of objects) is in a separate sheet. In this approach, the average path length from root to leaf is an anomaly measure of the corresponding object, since the leaves have outliers at earlier stages than the typical points.

Figure 1 illustrates that anomalies are more susceptible to isolation under random partitioning. We observe that the normal point x_i generally requires more separations for isolation. The x_0 anomaly point required fewer sections for isolation.

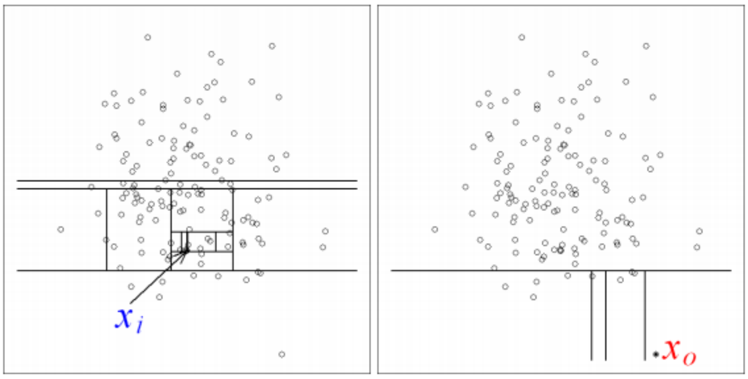

Figure 1. Isolated points $\boldsymbol{x_i}$ and $\boldsymbol{x_0}$

Anomalies are more susceptible to isolation and have a shorter path length. In the example above, x_i requires allocation of twelve random partitions (Figure 1, left); x_0 anomaly requires allocation of only four partitions (Figure 1, right). Since each section is randomly generated, individual trees are generated with different sets of sections. We average the path length across multiple trees to find the expected path length. Figure 2 shows that the average path lengths x_0 and x_i converge as the number of trees increases. When using 1000 trees, the average path lengths x_0 and x_i are 4.02 and 12.82, respectively. This shows that anomalies have a shorter path length than normal instances [2].

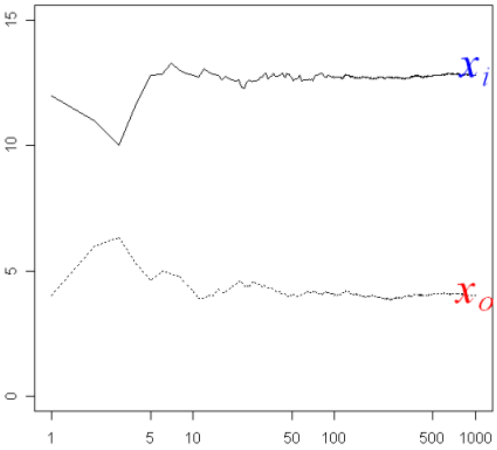

Figure 2. The average path lengths x_i and x_0 converge as the number of trees increases.

OSVM (One Class Support Vector Machine). OSVM model learns on data from a single class (usually a "normal" class). The trained model can be used to classify new data that is both similar and different from the training set. This is useful for detecting outliers, since there are usually few examples of outliers (anomalies), which makes it difficult to train a two-class classifier to distinguish them.

In order to get more universal decision boundaries by OSVM, kernel functions are often used to map the source object space to multidimensional spaces. Let *φ:X→Z* be a mapping from the source data space to the multidimensional object space. To train the OSVM model, we solve the following quadratic program:

$$\min_{\omega,\varepsilon_i,b} \frac{1}{2}\|\omega\|_2^2 + \frac{1}{vm}\sum_{i=1}^{m} \varepsilon_i - b \tag{1}$$

$$\forall_i = 1,2,\ldots\ldots,N$$

$$v \in (0,1], \varepsilon_i \geq 0,$$

$$\omega \cdot \varphi(z_i) \geq b - \varepsilon_i$$

where ε_i is a variable for the data point i. ν takes a value from 0 to 1; it limits a part of training errors from above and a part of support vectors from below. Consequently, a non-zero ν may create the OSVM model and exclude some of the training data as outliers from the normal class. To define belonging *x* to a normal class, we use the following function:

$$F(z) = sign\ (\omega \cdot \varphi(x) - b) = \begin{cases} +1\ normal, \\ -1\ outlier. \end{cases} \tag{2}$$

In this work, the kernels of radial basis functions (RBF) are selected. They do not take any parametric form of data distribution and as a result suit to capturing complex data.

3. The result of the methodology application

Let us examine the described methodology for finding anomalies in system logs using a special example. At the first stage we selected the popular GrayLog log file storage system, from which the csv file was uploaded for a specified period of time for further processing. The choice of the system was based on the fact that each event record was shown in raw form with the maximum possible information about the network packet. The studied upload per day from the GrayLog system was about 40 million records.

The second stage presented application of regular expressions during data preprocessing to remove unnecessary characters. Furthermore, empty rows were deleted, since their presence incorrectly affected further clustering. The data type "datetime64" was set for records containing data about the date and time of the event. The original domain names were assigned to second-level domains, because they could fall into different clusters during clustering when working with third-level and higher domains. The final step at this stage was to upload a smaller size to the final file.

At the third stage, data clustering process is aimed to identify atypical objects that cannot be attached to any cluster, or to simplify further data processing and decision-making by applying different analysis methods to each cluster.

Since we used K-means clustering algorithm of unsupervised learning, we had to select the number of clusters that would give us an optimal separation for further machine learning. The number of clusters was determined by iteration. The studied data set included 4 clusters.

At the fourth stage, abnormal records (domains) are detected using the described machine learning methods. The result of the algorithms is a csv file containing information about anomalous domains. The popular VirusTotal service was used for their subsequent evaluation and identification of malicious domains.

Machine learning algorithms revealed 84 Isolation forest and 96 OSVM anomalous domains. The result of domain verification is shown in Tables 1 and 2.

Table 1. Selection of malicious domains from abnormal Isolation forest domains.

№	Isolated Forest	Type
1	au79nt5wic4x.com	Malware
2	fbwat.ch	Phishing
3	rncdn7.com	Malicious
4	nbf9b5aurl.com	Malicious
5	reichelcormier.bid	Malicious
6	pubt.in	Malicious
7	m.me	Malicious
8	adsco.re	Malicious
9	aletorrenty.pl	Malware

Table 2. Selection of malicious domains from abnormal O-SVM domains.

№	OSVM	Type
1	pubt.in	Malicious
2	mgtracker.org	Malicious
3	ceoworldeventsummit.info	Spam
4	m.me	Malicious
5	au79nt5wic4x.com	Malware
6	nbf9b5aurl.com	Malicious
7	emlstart.com	Spam
8	aletorrenty.pl	Malware
9	klike.net	Malicious

This sample includes domains only with a negative rating on the VirusTotal site. Domains classified as "Suspicious" but having a neutral rating on VirusTotal were not entered. Furthermore, the domains that were not classified as malicious but had a negative rating from users of the VirusTotal service were not included. The domains of torrent trackers, sites with prohibited content and *.onion domains were also detected as a result of the algorithm, since they were prohibited on the corporate network.

The algorithm process identified 13 unique domains that were classified as malicious by the VirusTotal service.

4. Conclusion

This article discusses machine learning methods - Isolation forest and OSVM, used for anomalies search in log files. As a result, the following results were obtained:

- Raw log file of 40 million records was uploaded from the GrayLog system in "*.csv " format;
- Preprocessing of the resulting csv file was performed, which resulted the creation and processing of a data set of 21 million records;
- Selection of parameters for optimal data clustering using the K-means method;
- Search for anomalies in the prepared data set using Isolation Forest and OSVM methods;
- A manual check of "abnormal" domains for malicious ones using the VirusTotal service; identification of 13 unique domains found using machine learning methods.

This study shows the effectiveness of using machine learning methods in system log files analyzing. Existing security systems did not identify requests on domains previously, and the presence of these connections may pose a potential threat to the network infrastructure.

References

[1] Grubbs F E 1969 *American Society for Quality Procedures for Detecting Outlying Observations in Samples* vol 11

[2] Liu F T, Ting K M and Zhou Z H 2008 Isolation forest *Proceedings - IEEE International Conference on Data Mining, ICDM* pp 413–22

[3] Callegari C, Giordano S and Pagano M 2017 Entropy-based network anomaly Detection *2017 International Conference on Computing, Networking and Communications, ICNC 2017* (Institute of Electrical and Electronics Engineers Inc.) pp 334–40

[4] Kohout J, Komárek T, Čech P, Bodnár J and Lokoč J 2018 Learning communication patterns for malware discovery in HTTPs data *Expert Syst. Appl.* **101** 129–42

[5] Thomas T, Vijayaraghavan A P and Emmanuel S 2019 *Machine learning approaches in cyber security analytics* (Springer Singapore)

[6] Du M, Li F, Zheng G and Srikumar V 2017 DeepLog: Anomaly detection and diagnosis from system logs through deep learning *Proceedings of the ACM Conference on Computer and Communications Security* (Association for Computing Machinery) pp 1285–98

[7] Bertero C, Roy M, Sauvanaud C, Trédan G and Tredan G 2017 *Experience Report: Log Mining using Natural Language Processing and Application to Anomaly Detection*

[8] Hunt K 2016 *Log Analysis for Failure Diagnosis and Workload Prediction in Cloud Computing* (STOCKHOLM, SWEDEN)