electronics

Article

System Log Detection Model Based on Conformal Prediction

Yitong Ren [1], Zhaojun Gu [2,*], Zhi Wang [3], Zhihong Tian [4], Chunbo Liu [2], Hui Lu [4,*], Xiaojiang Du [5] and Mohsen Guizani [6]

1 College of Computer Science and Technology, Civil Aviation University of China, 300300 Tianjin, China; 2017052053@cauc.edu.cn
2 Information Security Evaluation Center, Civil Aviation University of China, 300300 Tianjin, China; luchamber@163.com
3 College of Cyber Science, Nankai University, 300071 Tianjin, China; zwang@nankai.edu.cn
4 Cyberspace Institute of Advanced Technology, Guangzhou University, 510006 Guangzhou, China; tianzhihong@gzhu.edu.cn
5 Department of Computer and Information Sciences, Temple University, Philadelphia, PA 19122, USA; dxj@ieee.org
6 Computer Science and Engineering Department, Qatar University, Doha 2713, Qatar; mguizani@gmail.com
* Correspondence: zjgu@cauc.edu.cn (Z.G.); luhui@gzhu.edu.cn (H.L.)

Received: 1 January 2020; Accepted: 27 January 2020; Published: 31 January 2020

Abstract: With the rapid development of the Internet of Things, the combination of the Internet of Things with machine learning, Hadoop and other fields are current development trends. Hadoop Distributed File System (HDFS) is one of the core components of Hadoop, which is used to process files that are divided into data blocks distributed in the cluster. Once the distributed log data are abnormal, it will cause serious losses. When using machine learning algorithms for system log anomaly detection, the output of threshold-based classification models are only normal or abnormal simple predictions. This paper used the statistical learning method of conformity measure to calculate the similarity between test data and past experience. Compared with detection methods based on static threshold, the statistical learning method of the conformity measure can dynamically adapt to the changing log data. By adjusting the maximum fault tolerance, a system administrator can better manage and monitor the system logs. In addition, the computational efficiency of the statistical learning method for conformity measurement was improved. This paper implemented an intranet anomaly detection model based on log analysis, and conducted trial detection on HDFS data sets quickly and efficiently.

Keywords: HDFS; anomaly detection; conformal prediction; confusion matrix

1. Introduction

As more and more things connect to the Internet of Things [1–4] (IoT), the amount of data associated with and generated by IoT devices grows exponentially. In the IoT ecosystem, log data are always an indispensable link as an important resource to understand system status. Because the Internet is more vulnerable to malicious code [5,6], there have been many research efforts in the current security threats and solutions facing the IoT [7–10]. They involve blockchain, machine learning, and edge computing. Compared with extranet logs, internal network system logs are often ignored. However, with the advent of the big data era, the amount of log data has exploded. Manual log analysis is no longer adequate to current requirements. Thus, automated detection technology in machine learning has become mainstream. With the increase of malicious behavior, once the internal network system fails, it may cause great economic losses and even personal information safety threats [11], especially in some highly exposed technical systems [12]. In order to ensure the log security, firstly, we need to parse unstructured text logs into structured ones. Then anomaly detection can be performed on the structured logs, which is also known as outlier detection. Its purpose is to find the processing that is different from the expected object. Among them, the method and model of anomaly log detection is the key to internet security, which has attracted a lot of attention from the industry.

Facing the era of big data, especially in the field of IoT, the fast and accurate analyses of the complex system log data have become a big challenge. Log mining tools [13,14] have been designed based on the context of system log words. Although log parsing is not required, large-scale log data processing takes a lot of time with poor accuracy. Based on the outlier detection method [15–19], syslog events appear in the period of weeks under certain circumstances. The outlier detection method excludes those normal system log events, and the calculation efficiency decreases with the increase of data amount, although some breakthroughs have been made in solving high-dimensional data [15–17].

In most of the system log anomaly detection methods, the whole process is divided into two steps: Log analysis and anomaly detection. Log parsing converts raw logs into structured logs in the form of log keys [20–23], extracting variables. Then researchers conduct historical logs clustering [24,25] to identify problems by using deep parse trees [26] based on log length or Principal Component Analysis (PCA) [27] for anomaly detection. In addition to unsupervised learning, there are some supervised learning methods [28]. Although these machine learning methods run very fast, they perform slightly worse in anomaly detection when data change because traditional machine learning for anomaly log detection only output simple predictions. In the field of deep learning [29], the neural network Long Short-Term Memory (LSTM) [30] method is used for modeling natural language sequences and performing anomaly detection from the system logs. However, it is time-consuming work.

In order to dynamically adapt to changing data, we use machine learning methods for anomaly detection from the perspective of statistical learning. Conformal Prediction (CP) [31,32] is a mathematical framework that classifies data using confidence. All classification or regression algorithms using fractional or real values can be transformed into conformal predictors. Therefore, CP can supplement the prediction results with effective confidence based on traditional machine learning techniques [33–35]. At present, CP algorithm has been well applied in the medical field [36], and Conformal Cluster (CC) [37,38] method exists in the clustering problem. We introduced the CP framework into system log anomaly detection so that system administrators can manage syslog data more effectively. Since all calculations of each test sample in the CP algorithm need to be restarted, the calculation efficiency is relatively low. Although inductive conformal prediction [39] improves the efficiency, it is limited by the choice of calibration set. We have improved the way of p-value calculation to achieve a balance between time cost and accuracy of anomaly detection.

Hadoop Distributed File System (HDFS) is the core subsystem of Hadoop, and is the foundation of data storage management in distributed computing. In the era of IoT big data, HDFS brings a lot of convenience to the application processing of the large data set. The direct or indirect losses caused by system problems are huge. CP framework is introduced to classify the system log data more accurately. This paper combines the anomaly detection method in machine learning with the statistical learning method in conformity measurement to solve the problem of log anomaly detection. Compared with the traditional classification methods based on static threshold in machine learning, the statistical learning method of conformity measure does not label the detected categories, but judges whether the log data are credible according to its confidence level. This method can dynamically adapt to the changing log data, improves the computation efficiency of CP, and reduces the time required to calculate the p-value. Finally, an intranet anomaly detection model is implemented based on log analysis. The statistical learning method of the conformity measure with higher computing efficiency is used for log data detection and detection result evaluation. The results show that the CP framework is effective in the problem of system log anomaly detection, which not only retains the fast and efficient detection of abnormal log data in machine learning, but also classifies the data more accurately. At the same time, this method improves the computational efficiency of CP.

The rest of this paper is organized as follows. Section 2 introduces the experimental framework, including log preprocessing, the method of anomaly detection, and evaluation metrics. Section 3 describes the impact of significance level and analyzes the experimental results. The final section summarizes the current study and proposes the future work.

2. Framework

This section describes the general framework of the experiment. The experimental model is first introduced, which mainly includes two main stages: Log preprocessing and anomaly detection. In the anomaly detection, the two machine learning algorithms used in the system log anomaly detection are presented, and then the CP framework is introduced and improved methods are proposed. At the end of this section, the experimental environment, experimental data, and model evaluation methods are presented.

In order to facilitate the comparison, in addition to the proposed *p*-value classification method, the threshold classification method is also given for comparison. The framework of the model is shown in Figure 1. The first stage of the model is log preprocessing. The raw log is obtained by log parsing and feature extraction to acquire the matrix of feature vector. The second stage is anomaly detection. In the threshold-based classification method, the machine learning model is trained to perform anomaly detection on the test set and obtain the probability until the measured data are normal for judgment. In the *p*-value-based classification method, the algorithm is regarded as a non-conformity function to obtain non-conformity scores and calculate the *p*-value. The maximum fault is set tolerance to detect anomalies.

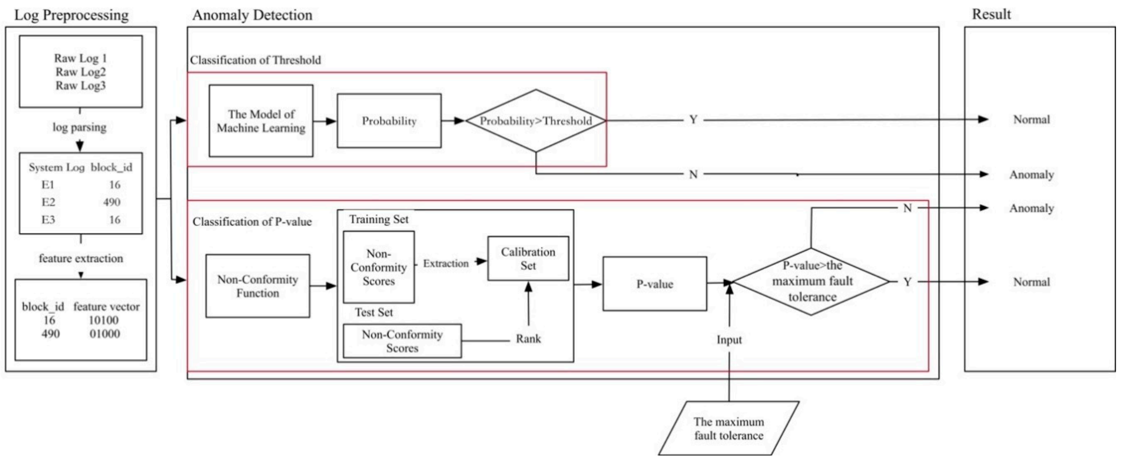


Figure 1. Framework of model.

2.1. *Log Preprocessing*

Log preprocessing consists of log parsing, feature extraction, and data cleansing. As shown in Table 1, log parsing resolves raw logs into structured logs. Through automatic log parsing, the variable portion of the raw log is replaced by a wildcard template, <*>. Feature extraction marks the raw log in the form of an event ID after it has been parsed into a structured log.

Table 1. Log preprocessing.

Raw Log	Structured Log	Block_ID	Event ID
PacketResponder 1 for block blk_16 terminating	PacketResponder <*> for block <*> terminating	16	E1
Verification succeeded for blk_490	Verification succeeded for <*>	490	E2
Verification succeeded for blk_16	Verification succeeded for <*>	16	E2

Every wildcard template corresponds to an event id, and it records the block ID that appears in each structured log. The structured logs are grouped into a log sequence according to the block ID, and the number of occurrences of each event ID in the log sequence is counted to form a feature vector. Multiple feature vectors are combined to obtain a feature vector matrix as the input of the machine learning model.

In this experiment, it was found that corresponding to the identical feature vectors in the data set, some block ID tags were different. In order to avoid the impact of these log data on the experiment, data cleaning was performed to remove these same feature vectors with different labels, a total of 550.

2.2. *Anomaly Detection*

The most important phase of the anomaly detection is the choice of algorithm. A commonly used supervised learning is to model the training set data with labels. The input test data set is analyzed and the results are obtained by establishing a model. In this paper, two supervised learning algorithms were used for experiments, i.e., Logistic Regression and Support Vector Machine (SVM).

Logistic Regression (LR) is widely used to monitor machine learning models and it often solves the problem of classification and regression. Data with two labels are separated as much as possible by fitting a line during the training. During the test, the feature vector of the unknown tag data is input to obtain the tag of the data. If the test data are farther from the fitted line, the probability of belonging to a certain type of tag is greater.

SVM is also a supervised learning model commonly used for classification and regression. Its goal is to use training data to fit a hyperplane in a high-dimensional space to separate the data of different labels, maximizing the spacing between the hyperplane and the nearest data points in different categories.

2.3. *Conformal Prediction*

Conformal Prediction (CP) determines the confidence level of test data by using training data as past experience. It can be used with algorithms, such as Bayesian, K-nearest neighbors, or supervised learning algorithms described above, to classify or cluster using fractional or real values. Table 2 gives a list of notations related to CP used in this paper.

Table 2. List of notations in the Conformal Prediction (CP).

Notation	Meaning	Notation	Meaning
D	Data Set	s_i	Non-conformity score of test sample data
C	Training Set	s_j	Non-conformity score of calibration set sample data
$C1$	Calibration Set	n	the total number of samples
A	Non-conformity Function	ε	the maximum fault tolerance /the significant level
z_i	test sample data	p	p-value
z_j	calibration set sample data	t_k	counter

In the Conformal Prediction, firstly the data set D, the training set C, $C \in D$, and a test sample data z are given, and a scoring method as the non-conformity measure function A is determined. The normal sample data in the training set C are all extracted to form a calibration set $C1 = \{z1, z2, ...z(n-1)\}$, $C1 \in C$. The calibration set calculates the non-conformity score along with the test set, and it is used to calculate the p-value of each sample of the test set, while these normal sample data are still retained in training set C.

A calibration set sample data or test sample data z are input to the non-conformity metric function A, outputting a value which is also called the non-conformity score s and indicates the inconsistency of the input. Then the Conformal Prediction uses the p-value to describe the percentile of the inconsistent score of a test sample data in a set of training samples, calculating the similarity from a statistical perspective. Every calibration set sample data z_j, $j \in (1, n\text{-}1)$ and the test sample data $z_{i,}$ which input non-conformity metric function A, will respectively obtain two non-conformity scores s_j and s_i. When calculating the p-value p_i of the test sample data, the sample data which are greater than or equal to the inconsistent score s_i in the statistical calibration set are divided by the total number of samples n. Finally, the maximum fault tolerance, i.e., the significant level ε is set. If $p_i \geq \varepsilon$, the sample is normal in the confidence level of $1-\varepsilon$.

Equation (1) calculates the non-conformity score s_i of the test sample z_i. Equation (2) calculates the p_i of the test sample.

$$s_i = A(C_1, z_i) \tag{1}$$

$$p_i = \frac{\left|\left\{j : s_i \geq s_{(n-1)}\right\}\right|}{n} \tag{2}$$

In the traditional threshold-based classification method, in order to judge the label of the test data, the logistic regression is classified by interval distribution. In the experiment, 0 means normal and 1 means anomaly. If the calculated probability $p < 0.5$, the data label is normal, and if the calculated probability $p > = 0.5$, the data are anomaly. Anomaly detection is performed by a linear support vector machine. If the test data are below the hyperplane, they are classified as normal. If they are above the hyperplane, they are classified as anomaly.

In the p-based classification method, two algorithms, i.e., logistic regression and support vector machine, are used as the consistency measurement function experiments in the consistency prediction algorithm. After obtaining the inconsistent scores from the original eigenvector matrix, the statistic p-value of each eigenvector is calculated according to the statistical analysis of the calculation results. The category of the log data is determined according to the size relationship and credibility of the given p-value. Since the probability that the obtained label set contains the real label can be given, it makes the prediction result credible.

2.4. Improving p-Value Calculation Time

The non-conformity score obtained from the calibration set sample data is used as an empirical sequence, and the non-conformity score of each data in the test set is calculated, of which percentile in the empirical sequence is taken as the p-value. However, such calculations are inefficient facing big data. It has been verified that the Inductive Conformal Prediction (ICP) improves the efficiency of calculating p-values. When analyzing the log data of medium-sized data in this way, however, the analyzing accuracy is limited by the extracted calibration set because of the accounts of calibration set for 20% of the training set data volume.

Therefore, when dealing with medium size of log data, we use the representation of the non-conformity score count. Different eigenvector types may get the same non-conformity score, but when calculating the p-value, the percentile of the test set data in the empirical sequence is only related to the non-conformity score. Therefore, as shown by Algorithm 1, when the non-conformity score s_j of the calibration set data is calculated, each time the same score s_k occurs, the counter t_k of the score is incremented by one. Using the representation of occurrence number of the non-conformity score, we calculate the percentile of the test set data in the empirical sequence, while the empirical sequence at this time contains only the non-conformity scores of the different scores. This will greatly reduce the p-value calculation time and improve the calculation efficiency.

Algorithm 1. The *p*-value calculation time reduction procedure.

Require: non-conformity score of test sample s_i, non-conformity scores set of calibration *list*, the number of list n
Result: *p*-value p_i

```
Count list-new ← list and p-v ← 0
Sort scores from small to large list-sort ← list-new
For j in list-sort do
    if $j < s_i$ then
        p-v ← p-v + list-new[j]
    else
        break
$p_i = 1 - (p\text{-}v/(n+1))$
```

2.5. Evaluation Metrics

The experiments were conducted on the following platform: A 3.1 GHz Intel Core i5 processor, 8 GB RAM, and MacOS operating system. Using the confusion matrix as the evaluation metrics of the experimental results of the supervised learning model, the numbers of True Positive(TP), True Negative(TN), False Positive(FP), and False Negative(FN) in the confusion matrix were obtained. Then accuracy, specificity, negative predictive value (NPV), precision, recall/sensitivity, and F1 indicators were used to show more intuitive results. Finally, the misclassification data were traced by case analysis, in order to summarize the eigenvector type of the misjudgment data and to find out the cause of misjudgment. Equations (3)–(8) respectively show the calculation method of accuracy, specificity, NPV, precision, recall, and F1 score.

$$\text{Accuracy} = \frac{TP + TN}{TP + FP + FN + TN} \tag{3}$$

$$\text{Specificity} = \frac{TN}{FP + TN} \tag{4}$$

$$\text{NPV} = \frac{TN}{TN + FN} \tag{5}$$

$$\text{Precision} = \frac{TP}{TP + FP} \tag{6}$$

$$\text{Recall} = \frac{TP}{TP + FN} \tag{7}$$

$$\text{F1} = \frac{2 * Precision * Recall}{(Precision + Recall)} \tag{8}$$

The Hadoop Distributed File Management System (HDFS) is an implementation of the Hadoop abstract file system. The 1.58 G HDFS_1 data set with a total of 11,175,629 log messages used in this experiment was provided by the Chinese University of Hong Kong. In addition, there is a sample data set with a 365,298 Block_ID labels. The HDFS_1 data set contained each block_id and its events of addition, movement, deletion, and so on for nearly 39 hours, and was marked with the event id. The HDFS_1 data set was recorded in chronological order and divided into training and test sets in a ratio of 8:2. In the two split data sets, each unique block_id contained one or more event types; and in the block_id, each event type appeared once, multiple times, or not. All events were sorted into a log sequence by event id. The log sequence was counted according to the event id number from small to large for the number of occurrences of each event. The obtained result is called a feature vector. Each segmented data set resulted in a feature vector matrix. The data set was labeled for normal or anomaly condition of block id. After data cleaning, each feature vector in the training set was matched with the tagged block_id sample data. So, we obtained 459,608 labeled training set data, including 14,844 anomaly sample data. While in the test set, 114,903 pieces of data were marked as normal, and 1675 pieces of data were marked as anomaly.

3. Experiments and Results

This section describes the experimental process and analyzes the experimental results. The section is divided into four parts. The influence of the significance on the experiment is introduced first. Then it shows the experimental results according to the evaluation metrics and the results of time cost before and after the optimization of the method. Finally, the experimental results are analyzed.

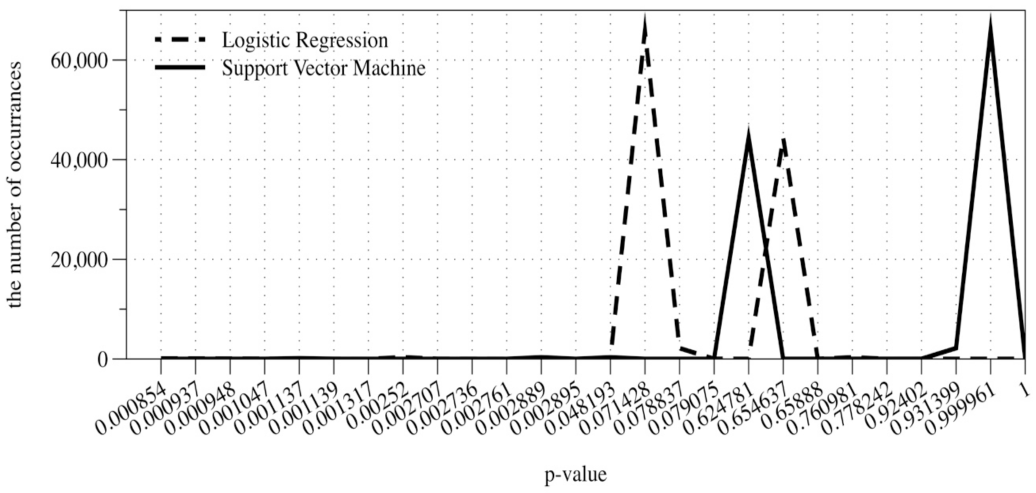
3.1. The Impact of the Significance Level on the Experiment

In order to better understand the significance level in the conformal prediction algorithm, we calculated the *p*-values of the two detection models and found that the difference between the normal log data and the anomaly log data was large. Figure 2a shows the relationship between the *p*-value of the normal log data in the test set and the number of occurrences. Figure 2b shows the relationship between the *p*-value of the anomaly log data in the test set and the number of occurrences. The number of occurrences of normal log data increased with the increase of the *p*-value, while the number of occurrences of the anomaly log data decreased as the *p*-value increased. The significance level of the normal log data was mostly concentrated after 0.0028, while the significance level of the anomaly log data was very close to zero. When selecting a calibration set sample, all selected samples were normal. Therefore, it can be said that the smaller the *p*-value was, the more significant the difference between the training set sample and the calibration set sample was. However, the training set and calibration set cannot be perfect. In offline log anomaly detection analysis, the training set and calibration set did not include all normal or anomaly conditions. In the case that all samples of the test calibration set were normal, we concluded that the smaller the *p*-value of the normal sample was, the more significant the difference between it and the sample in the calibration set was, and the higher the probability of being classified as anomaly became.

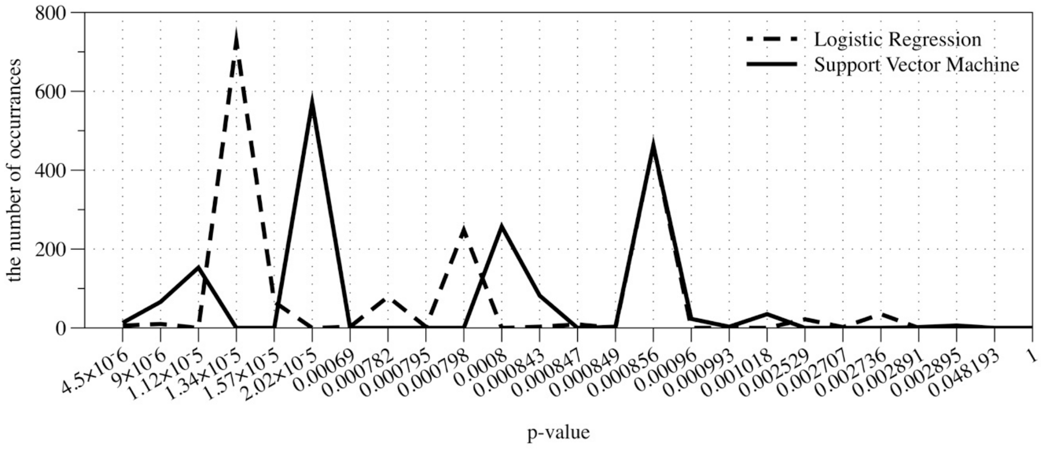
The maximum fault tolerance probability of the two algorithms was set to be 0.999, that is, the significance level was 0.001. Taking the Logistic Regression (LR) as an example, if the *p*-value was greater than the significant level of 0.001, it was considered that the difference was not obviously at the 99.9% significance level, which was normal. Otherwise, the difference was obvious and anomaly. If the maximum fault tolerance probability was changed to 0.99, that is, the significance level was 0.01, and when the *p*-value was greater than the significant level of 0.01, it was considered that the difference was not obviously at the 99% significance level, which was normal. Otherwise, the difference was obvious and anomaly. It can be seen from the comparison of these two different levels of significance values that although the number of FNs increased, the number of FPs decreased significantly, and both the recall rate and the F1 value increased. Unlike traditional threshold-based, considering the way in which the detection category labels were given, the *p*-based approach dynamically changed the level of significance to accommodate constant changes in the data. It is no longer a matter of dividing data by category label only, but using a significant level to judge whether the difference is obvious or not, thereby judging whether the piece of data is credible.

3.2. Comparison of Results

Tables 3 and 4 show the results using LR based on threshold and based on *p*-value experiments, and the maximum fault tolerance probability is set to 0.991, the level of significance was 0.009. Tables 5 and 6 show experimental results using threshold-based and *p*-based support vector machine (SVM), and the maximum fault tolerance probability was set to 0.99896, so the significance level was 0.00104. It can be seen from the results that the CP framework was still robust in the system log anomaly detection problem. Compared with the traditional threshold-based classification method, the classification result based on *p*-value was not only successfully applied effectively, but also had some improvements. In the LR, the number of false negative (FN) was reduced, and the NPV, precision, and F1 values were improved, while in the SVM, the number of false positive (FP) was reduced, and the specificity, recall, and F1 values were improved.



(**a**)



(**b**)

Figure 2. (**a**) Shows the relationship between the *p*-value of the normal log data in the test set and the number of occurrences. (**b**) Shows the relationship between the *p*-value of the anomaly log data in the test set and the number of occurrences. Normal logs are concentrated in areas with large *p*-values, while abnormal logs have very small *p*-values.

Table 3. Comparison of confusion matrix based on Logistic Regression (LR) results.

Evaluation Metrics	Based on Threshold	Based on *p*-Value
TP	113,094	113,151
TN	1,616	1,616
FP	59	59
FN	134	77

Table 4. Performance result based on LR.

Evaluation Metrics	Based on Threshold	Based on *p*-Value
Accuracy	0.998	0.999
Specificity	0.964	0.964
NPV	0.923	0.954
Precision	0.923	0.955
Recall	0.965	0.965
F1	0.944	0.960

Table 5. Comparison of confusion matrix based on Support Vector Machine (SVM) results.

Evaluation Metrics	Based on Threshold	Based on *p*-Value
TP	113,228	113,228
TN	1,606	1,664
FP	69	8
FN	0	0

Table 6. Performance result based on SVM.

Evaluation Metrics	Based on Threshold	Based on *p*-Value
Accuracy	0.999	1.0
Specificity	0.958	0.995
NPV	1.0	1.0
Precision	1.0	1.0
Recall	0.959	0.995
F1	0.979	0.998

3.3. Comparison of Calculation Efficiency

In the CP framework, data ranking needs to be calculated for each measured sample data according to past experience when calculating the *p*-value, which has long time consumed and low efficiency. Some improvements were illustrated in Section 2.4. In this part, we only recorded the time that the test set data used to calculate the *p*-value. System behavior of HDFS data is relatively simpler, and there are many similar data; meanwhile, the inconsistent scores are similar or even the same. In this case, the CP took a lot of time to calculate the *p*-value. LR and SVM spent, respectively, 2637 s and 601 s in the CP. After modification of the *p*-value calculating method in Section 2.4, the two anomaly detection algorithms finished the calculation, respectively, in 10 s and 4 s, the calculation time was significantly shortened.

3.4. Analysis of Results

By analyzing the data set of training phase and test phase, some normal feature vectors which never appeared before in the training set were found. While in the LR, the conformity scores of these newly appeared feature vectors were very low, locating at the boundary of the normal feature vector and the anomaly feature vector. LR was a generalized linear regression analysis. If the level of significance was further reduced, although the number of false positives can be reduced, the number of false negatives was greatly increased. For example, the level of significance 0.000854384 was less than 0.000856632, but the number of normal log data was respectively 76 and 0, and the number of anomaly log data was 1 and 462 in the false negative. This is the reason for 77 false negatives in the LR. In the same way, the same situation occurred between the significance level 0.0025 and 0.0028, from which appeared 59 false positives data. These log data were obviously at the boundary of the normal feature vector and the anomaly feature vector. The SVM in this experiment did not have such a phenomenon, because SVM classification method was to fit a hyperplane in a high-dimensional space to separate the data of different labels. When analyzing the false positives of the two algorithms, it was found that the feature vectors of false positives were very similar and had few types. For example, there were 59 false positives in the LR, but only three feature vector types were included. The difference between the two feature vector types was only the number of occurrences of event 5, which were, respectively, 4 and 5 times. By the same token, if the level of significance increased, the number of false positives also increased, although the number of false negatives decreased. It is important to choose the appropriate level of significance.

We compared the false negative and false positive of the two algorithms and found that only one eigenvector type appeared in the false negative of both algorithms. Other misclassifications only occurred once in both algorithms. So, we can conclude that different algorithms had different learning strategies, and the perspectives on test data were also different. If we use the appropriate algorithm integration or ensemble learning, the number of false negatives and false positives can be further reduced. This is what we are planning for later papers.

4. Conclusions and Future Work

This paper combined statistical learning methods and machine learning to detect and analyze the anomaly of HDFS logs. Based on machine learning algorithms for anomaly detection, the framework of CP was used as a statistical learning method for conformity measurement. It was improved to balance the accuracy of the experimental results and the efficiency of time cost of intranet anomaly detection model based on log analysis. Compared with the traditional threshold-based classification method, the p-value-based classification method in the CP was no longer a classification of 0 and 1. It scored by means of conformity measures and obtained the percentiles of the score in the overall sample. It dynamically adapted to the constantly changing anomalous data by adjusting the maximum fault tolerance probability. Optimization was performed by using the method of score corresponding to the number of occurrences, which overcame the disadvantage of inefficient calculation of p-value in the CP. The experiments successfully proved that the CP framework is still effective in the anomaly detection of system logs. Through experiments, we not only compared the two classification methods, but also made data analysis and realized quick and accurate anomaly detection on log data.

In the future, we will incorporate more machine learning algorithms into the CP framework for anomaly detection of system logs based on the current research. Each algorithm has its own learning strategy, and Ensemble Learning can combine these machine learning algorithms. The diversity of models and the accuracy of results are the goals of Ensemble Learning. But each algorithm has its own advantages and disadvantages, and the performance of the algorithm model may conflict in some cases. Choosing the proper basic algorithm in Ensemble Learning is a challenge. So, if we can make some breakthroughs in Ensemble Learning, the appearance of abnormal data which cannot be detected will be greatly reduced. At the same time, the overfitting of the model also needs to be considered. As the volume of the model increases, we will consider using edge computing [40–42] to further reduce the time overhead. Selection and weight distribution of Ensemble Learning algorithm are focuses of our future work.

Author Contributions: Conceptualization, Y.R., Z.W., and C.L.; methodology, Y.R.; validation, Y.R.; formal analysis, Y.R.; supervision, Z.G., Z.T., H.L., X.D., and M.G. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Guangdong Province Key Area R&D Program of China under Grant No. 2019B010137004; the National Natural Science Foundation of China under Grant No.61871140, No. U1636215, and No. 61972108; the National Key Research and Development Plan under Grant No. 2018YFB0803504; Civil Aviation Safety Capacity Building Project; and Guangdong Province Universities and Colleges Pearl River Scholar Funded Scheme (2019).

Acknowledgments: The author would like to thank the Chinese University of Hong Kong for providing the HDFS log data.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Yin, L.; Luo, X.; Zhu, C.; Wang, L.; Xu, Z.; Lu, H. ConnSpoiler: Disrupting C&C Communication of IoT-Based Botnet through Fast Detection of Anomalous Domain Queries. *IEEE Trans. Ind. Inform.* **2019**, *16*, 1373–1384. [CrossRef]
2. Qiu, J.; Du, L.; Zhang, D.; Su, S.; Tian, Z. Nei-TTE: Intelligent Traffic Time Estimation Based on Fine-grained Time Derivation of Road Segments for Smart City. *IEEE Trans. Ind. Inform.* **2019**, *16*, 2659–2666. [CrossRef]
3. Tian, Z.; Gao, X.; Su, S.; Qiu, J. Vcash: A Novel Reputation Framework for Identifying Denial of Traffic Service in Internet of Connected Vehicles. *IEEE Internet Things J.* **2019**, 1. [CrossRef]

4. Wu, L.; Du, X.; Wang, W.; Lin, B. An Out-of-band Authentication Scheme for Internet of Things Using Blockchain Technology. In Proceedings of the International Conference on Computing, Networking and Communications, Maui, HI, USA, 5–8 March 2018.
5. Xiao, L.; Li, Y.; Huang, X.; Du, X. Cloud-based Malware Detection Game for Mobile Devices with Offloading. *IEEE Trans. Mob. Comput.* **2017**, *16*, 2742–2750. [CrossRef]
6. Dong, P.; Du, X.; Zhang, H.; Xu, T. A Detection Method for a Novel DDoS Attack against SDN Controllers by Vast New Low-Traffic Flows. In Proceedings of the 2016 IEEE International Conference on Communications (ICC), Kuala Lumpur, Malaysia, 22–27 May 2016.
7. Tsiropoulou, E.E.; Baras, J.S.; Papavassiliou, S.; Qu, G. On the Mitigation of Interference Imposed by Intruders in Passive RFID Networks. In *International Conference on Decision and Game Theory for Security*; Springer: Cham, Switzerland, 2016; pp. 62–80.
8. Zhang, K.; Liang, X.; Lu, R.; Shen, X. Sybil Attacks and Their Defenses in the Internet of Things. *IEEE Internet Things J.* **2014**, *1*, 372–383. [CrossRef]
9. Sajid, A.; Abbas, H.; Saleem, K. Cloud-assisted IoT-based SCADA systems security: A review of the state of the art and future challenges. *IEEE Access* **2016**, *4*, 1375–1384. [CrossRef]
10. Hassija, V.; Chamola, V.; Saxena, V.; Jain, D.; Goyal, P.; Sikdar, B. A Survey on IoT Security: Application Areas, Security Threats, and Solution Architectures. *IEEE Access* **2019**, *7*, 82721–82743. [CrossRef]
11. Yang, G.; Jiang, M.; Ouyang, W. IoT-based remote pain monitoring system: From device to cloud platform. *IEEE J. Biomed. Health Inform.* **2018**, *22*, 1711–1719. [CrossRef] [PubMed]
12. Vamvakas, P.; Tsiropoulou, E.E.; Papavassiliou, S. Exploiting prospect theory and risk-awareness to protect UAV-assisted network operation. *EURASIP J. Wirel. Commun. Netw.* **2019**, *1*, 286. [CrossRef]
13. Vaarandi, R. A Data Clustering Algorithm for Mining Patterns from Event Logs. In Proceedings of the 2003 IEEE Workshop on IP Operations and Management, Kansas City, MO, USA, 3 October 2003; pp. 119–126.
14. Makanju, A.A.O.; Zincir-Heywood, A.N.; Milios, E.E. Clustering event logs using iterative partitioning. In Proceedings of the International Conference on Knowledge Discovery and Data Mining, Paris, France, June 2009. [CrossRef]
15. Angiulli, F.; Pizzuti, C. Fast outlier detection in high dimensional spaces. In Proceedings of the European Conference on Principles of Data Mining and Knowledge Discovery, Helsinki, Finland, 19–23 August 2002; pp. 15–27.
16. Zimek, A.; Schubert, E.; Kriegel, H.-P. A survey on unsupervised outlier detection in high-dimensional numerical data. Stat. Anal. Data Min. *ASA Data Sci. J.* **2012**, *5*, 363–387.
17. Pang, G.; Cao, L.; Chen, L.; Liu, H. Learning Representations of Ultrahigh-dimensional Data for Random Distance-based Outlier Detection. In Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data mining, London, UK, August 2018. [CrossRef]
18. Breunig, M.M.; Kriegel, H.-P.; Ng, R.T.; Sander, J. LOF: Identifying density-based local outliers. In Proceedings of the ACM International Conference on Management of Data and Symposium on Principles of Database Systems, Dallas, Texas, USA, May 2000; pp. 93–104.
19. Ramaswamy, S.; Rastogi, R.; Shim KAIST, K. Efficient algorithms for mining outliers from large data sets. In Proceedings of the ACM International Conference on Management of Data and Symposium on Principles of Database Systems, Dallas, Texas, USA, May 2000; pp. 427–438.
20. Fu, Q.; Lou, J.-G.; Wang, Y.; Li, J. Execution anomaly detection in distributed systems through unstructured log analysis. In Proceedings of the 2009 Ninth IEEE International Conference on Data Mining, Miami, FL, USA, 6–9 December 2009; pp. 149–158.
21. Tang, L.; Li, T.; Perng, C.-S. LogSig: Generating system events from raw textual logs. In Proceedings of the 20th ACM international conference on Information and knowledge management, Glasgow Scotland, UK, October 2011; pp. 785–794.
22. Du, M.; Li, F. Spell: Streaming Parsing of System Event Logs. In Proceedings of the 2016 IEEE 16th International Conference on Data Mining (ICDM), Barcelona, Spain, 12–15 December 2016; pp. 859–864.
23. He, P.; Zhu, J.; He, S.; Li, J.; Lyu, M.R. An evaluation study on log parsing and its use in log mining. In Proceedings of the 2016 46th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN), Toulouse, France, 28 June–1 July 2016; pp. 654–661.

24. Zong, B.; Song, Q.; Martin, R.M.; Wei, C.; Lumezanu, C.; Cho, D.; Haifeng, C. Deep autoencoding gaussian mixture model for unsupervised anomaly detection. In Proceedings of the 6th International conference on Learning Repretations, Vancouver, BC, Canada, 30 April–3 May 2018.
25. Lin, Q.; Zhang, H.; Lou, J.-G.; Zhang, Y.; Chen, X. Log Clustering Based Problem Identification for Online Service Systems. In Proceedings of the 2016 IEEE/ACM 38th International Conference on Software Engineering Companion (ICSE-C), Austin, TX, USA, 14–22 May 2016; pp. 102–111.
26. He, P.; Zhu, J.; Zheng, Z.; Lyu, M.R. Drain: An online log parsing approach with fixed depth tree. In Proceedings of the 2017 IEEE International Conference on Web Services, Honolulu, HI, USA, 25–30 June 2017; pp. 33–40.
27. Xu, W.; Huang, L.; Fox, A.; Patterson, D.; Jordan, M. Online system problem detection by mining patterns of console logs. In Proceedings of the IEEE International Conference on Data Mining (ICDM), Miami, FL, USA, 6–9 December 2009; pp. 588–597.
28. He, S.; Zhu, J.; He, P.; Lyu, M.R. Experience Report: System Log Analysis for Anomaly Detection. In Proceedings of the International Symposium on So ware Reliability Engineering (ISSRE), Ottawa, ON, Canada, 23–27 October 2016; pp. 207–218.
29. Tian, Z.; Luo, C.; Qiu, J.; Du, X.; Guizani, M. A Distributed Deep Learning System for Web Attack Detection on Edge Devices. *IEEE Trans. Ind. Inform.* **2020**, *16*, 1963–1971. [CrossRef]
30. Du, M.; Li, F.; Zheng, G.; Srikumar, V. DeepLog: Anomaly Detection and Diagnosis from System Logs Through Deep Learning. In Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, Dallas, Texas, USA, October 2017; pp. 1285–1298.
31. Vovk, V.; Gammerman, A.; Saunders, C. Machine learning applications of algorithmic randomness. In Proceedings of the 16th International Conference on Machine Learning (ICML'99), Bled, Slovenia, 27–30 June 1999; pp. 444–453.
32. Vork, V.; Gammerman, A.; Shafer, G. *Algorithmic Learning in a Random World*; Springer: New York, NY, USA, 2005.
33. Johansson, U.; Boström, H.; Löfström, T. Conformal Prediction Using Decision Trees. In Proceedings of the 2013 IEEE 13th International Conference on Data Mining, Dallas, TX, USA, 7–10 December 2013.
34. Papadopoulos, H.; Vovk, V.; Gammerman, A. Regression Conformal Prediction with Nearest Neighbours. *J. Artif. Intell. Res.* **2011**, *40*, 815–840. [CrossRef]
35. Johansson, U.; Boström, H.; Löfström, T.; Linusson, H. Regression conformal prediction with random forests. *Mach. Learn.* **2014**, *97*, 155–176.
36. Cortés-Ciriano, I.; Bender, A. Andreas Bender; Concepts and Applications of Conformal Prediction in Computational Drug Discovery. *arXiv* **2019**, arXiv:1908.03569.
37. Cherubin, G.; Nouretdinov, I.; Gammerma, A. Conformal Clustering and Its Application to Botnet Traffic. In Proceedings of the International Symposium on Statistical Learning and Data Sciences, Egham, UK, 13–20 April 2015; pp. 313–322.
38. Nouretdinov, I.; Gammerman, J.; Fontana, M.; Rehal, D. Multi-level conformal clustering: A distribution-free technique for clustering and anomaly detection. *arXiv* **2019**, arXiv:1910.08105. [CrossRef]
39. Papadopoulos, H.; Proedrou, K.; Vovk, V.; Gammerman, A. Inductive confidence machines for regression. In Proceedings of the 2002 European Conference on Machine Learning (ECML), Helsinki, Finland, 19–23 August 2002; pp. 345–356.
40. Shen, M.; Ma, B.; Zhu, L.; Mijumbi, R.; Du, X.; Hu, J. Cloud-Based Approximate Constrained Shortest Distance Queries over Encrypted Graphs with Privacy Protection. *IEEE Trans. on Inf. Forensics Secur.* **2018**, *13*, 940–953. [CrossRef]
41. Tian, Z.; Shi, W.; Wang, Y.; Zhu, C.; Du, X.; Su, S.; Sun, Y.; Nadra, G. Real Time Lateral Movement Detection based on Evidence Reasoning Network for Edge Computing Environment. *IEEE Trans. Ind. Inf.* **2019**, *15*, 4285–4294. [CrossRef]
42. Xiao, L.; Wan, X.; Dai, C.; Du, X.; Chen, X.; Guizani, M. Security in Mobile Edge Caching with Reinforcement Learning. *IEEE Wirel. Commun.* **2018**, *25*, 116–122. [CrossRef]