Hindawi
Wireless Communications and Mobile Computing
Volume 2018, Article ID 4680867, 10 pages
https://doi.org/10.1155/2018/4680867

Research Article

Intrusion Detection System Based on Decision Tree over Big Data in Fog Environment

Kai Peng,[1] **Victor C. M. Leung**,[2] **Lixin Zheng**,[1,3] **Shangguang Wang**,[4] **Chao Huang**,[1] **and Tao Lin**[1]

[1] *College of Engineering, Huaqiao University, Quanzhou, Fujian 362021, China*
[2] *Department of Electrical and Computer Engineering, The University of British Columbia, Vancouver, BC, Canada V6T 1Z4*
[3] *Fujian Provincial Academic Engineering Research Centre in Industrial Intellectual Techniques and Systems, Quanzhou, Fujian 362021, China*
[4] *State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing 100876, China*

Correspondence should be addressed to Kai Peng; pkbupt@gmail.com

Received 6 December 2017; Accepted 4 February 2018; Published 6 March 2018

Academic Editor: Xuyun Zhang

Fog computing, as the supplement of cloud computing, can provide low-latency services between mobile users and the cloud. However, fog devices may encounter security challenges as a result of the fog nodes being close to the end users and having limited computing ability. Traditional network attacks may destroy the system of fog nodes. Intrusion detection system (IDS) is a proactive security protection technology and can be used in the fog environment. Although IDS in tradition network has been well investigated, unfortunately directly using them in the fog environment may be inappropriate. Fog nodes produce massive amounts of data at all times, and, thus, enabling an IDS system over big data in the fog environment is of paramount importance. In this study, we propose an IDS system based on decision tree. Firstly, we propose a preprocessing algorithm to digitize the strings in the given dataset and then normalize the whole data, to ensure the quality of the input data so as to improve the efficiency of detection. Secondly, we use decision tree method for our IDS system, and then we compare this method with Naïve Bayesian method as well as KNN method. Both the 10% dataset and the full dataset are tested. Our proposed method not only completely detects four kinds of attacks but also enables the detection of twenty-two kinds of attacks. The experimental results show that our IDS system is effective and precise. Above all, our IDS system can be used in fog computing environment over big data.

1. Introduction

Fog computing [1, 2] was defined as a highly virtualized computing platform for migrating cloud computing center tasks to network edge devices. Fog computing provides computing, storage, and networking service between mobile users and traditional Cloud platform, which is complementary to Cloud. The fog computing introduces the middle layer between the cloud and the mobile users, extending the cloud-based network architecture [3–6]. A basic fog framework is shown in Figure 1, each mobile user is connected to one of the fog nodes. Meanwhile, fog nodes could be interconnected with each other and are linked to the Cloud [7]. The fog computing reduces unnecessary multiple communication between the cloud computing center and the mobile users. For instance, when the number of users has increased dramatically, these users can obtain the service by visiting the contents of the cache in the fog servers so as to reduce network delay [8]. And it also significantly reduces the bandwidth of the backbone link load [9, 10]. Unfortunately, the nodes in fog environment are close to the mobile users, and fog computing nodes are usually composed of devices with weak computing ability. Traditional network attacks are widely presented in fog environment; fog devices may face network security challenges. However, Intrusion Detection Systems (IDS) can be used for fog environment [11].

IDS is designed to ensure network security and the main task is detect malicious activities of the host or network

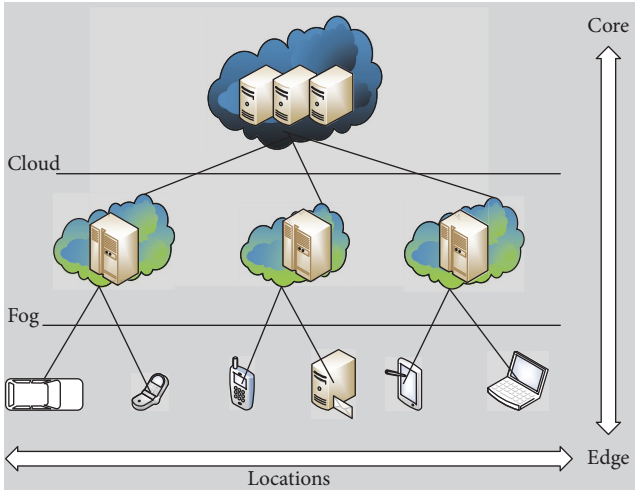

Figure 1: Fog between edge and cloud.

and then respond in a timely manner [12]. The definition of intrusion detection was first formally described in the 1980s [13]. In addition, the concept of real-time anomaly detection was proposed by Denning [14]. Pattern matching algorithm is one of the core technologies of IDS. Misuse detection based on AC, BM, MWM, and other matching algorithms [15] can make IDS have a passive detection of known attacks. However, modern attacks are increasingly inclined to form an unknown intrusion technology by integrating a variety of known intrusion technology. Meanwhile, improved IDS methods usually take proactive protection based on deviation detection and user behavior anomaly detection. For instance, statistical model, Bayesian reasoning, and cluster analysis [16] can make up for the lack of pattern matching, so that the system has a certain detection of unknown attacks. KNN algorithm [17] is widely used in pattern recognition, classification, and regression. Same as KNN, vector automatic classification algorithms, support vector machine [18–20], neural network algorithm [21], Bayesian algorithm [22–24], and K means algorithm are also widely used for IDS [25, 26].

Although the IDS in tradition network has been well investigated, unfortunately directly use them in fog computing environment may not inappropriate. Fog nodes produce massive amounts of data at all times, and, thus, enabling an IDS system over big data in fog environment is of paramount importance. More specifically, the existing researches mainly present the experiments on 10% KDDCUP99 dataset [27]. Although these methods have achieved good results, we cannot judge their efficiency when they are presented in the big data environment, even in the full dataset. In addition, there are four classification methods for network attacks, and also twenty-two classification methods in KDDCUP99. However, the existing research mainly focuses on the detection precision of four attacks but did not consider the detection of twenty-two attacks.

In order to address the above issue, we propose an IDS system based on decision tree over Anaconda [28]. Firstly, we propose a preprocessing algorithm to digitize the strings in the given dataset and then normalize the whole data, to ensure the quality of the input data so as to improve the efficiency of detection. Secondly, we use decision tree method for the detection of network attacks in our proposed IDS system, and then we compare this method with Naïve Bayesian method as well as KNN method. More specifically, three modes of Naïve Bayesian method are compared. And the experiment results show that our proposed IDS system is precise.

Our contributions in this study can be summarized as follows.

(1) For one thing, both the 10% dataset and the full dataset are tested in our IDS system, which proves that our IDS system is effective for big data environment.

(2) For another, we not only complete the detection of four kinds of attacks but also implement the detection of twenty-two kinds of attacks. The results show that our IDS system has a higher detection coverage of network attacks.

(3) In addition, the calculation time of each method is compared. To ensure the detection accuracy, although the calculation time of decision tree is not the best one, the time is also acceptable and can be used for big data environment.

The rest of the paper is organized as follows. In Section 2, the preliminaries are introduced. Section 3 specifies our proposed IDS system. The experimental evaluation is described in Section 4. Section 5 presents the related work. Finally, we conclude our work and describe the future work in Section 6.

2. Preliminaries

In this section, we firstly introduce the problem model and relevant formulas in Section 2.1 and then introduce the evaluation indicators of IDS detection in Section 2.2.

2.1. Problem Model and Relevant Formulas. The object of decision tree is to construct a decision tree model based on a given dataset to enable it to classify the new instances correctly. There are many methods to construct the decision tree, such as ID3 and C4.5 [29] and CART (Classification and Regression Trees) [30, 31]. In this study, we will use CART over Anaconda [28] for our IDS system. The relevant formulas are shown as follows.

For given dataset $T = \{(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)\}$, where

$$x_i = \left(x_i^{(1)}, x_i^{(2)}, \ldots, x_i^{(n)}\right)^T, \quad i = 1, 2, \ldots, N, \tag{1}$$

x_i is the input instance and represents a network packet record. x_i has n features. N indicates the number of records of the packets contained in the dataset T. $y_i \in \{0, 1, 2, \ldots, K-1\}$ is the class tag which means the result of each detection record.

Let Q represent the data at node m, where X_m is the training data in node m.

For each split $\theta = (j, t_m)$ which consists of a feature j and a threshold t_m, the data is divided into two subsets of $Q_1(\theta)$ and $Q_2(\theta)$:

$$Q_1(\theta) = (x, y) \mid x_j \le t_m,$$
$$Q_2(\theta) = Q \setminus Q_1(\theta). \tag{2}$$

The impurity of m can be obtained by using an impurity function $H()$:

$$G(Q, \theta) = \frac{n_1}{N_m} H(Q_1(\theta)) + \frac{n_2}{N_m} H(Q_2(\theta)). \tag{3}$$

Select the parameter to minimize the impurity:

$$\theta^* = \arg\min_{\theta} G(Q, \theta). \tag{4}$$

Recourse for both $Q_1(\theta^*)$ and $Q_2(\theta^*)$ until N_m reaches the maximum depth and thus $N_m < \min_{\text{samples}}$ or $N_m = 1$.

For the classification of IDS, $y_i \in \{0, 1, 2, \ldots, K-1\}$ for node m represents a region of R_m with instances of N_m.

Assume that p_{mk} is the proportion of class k instance in m and can be obtained by the following formula:

$$p_{mk} = \frac{1}{N_m \sum_{xj \in R_m} I(y_i = k)}. \tag{5}$$

The measure of impurity is generally named as Gini and can be obtained by the following formula:

$$H(X_m) = -\sum_k p_{mk}(1 - p_{mk}). \tag{6}$$

Cross entropy can be obtained by the following formula:

$$H(X_m) = -\sum_k p_{mk} \log(p_{mk}). \tag{7}$$

Misclassification can be obtained by the following formula:

$$H(X_m) = 1 - \max(p_{mk}). \tag{8}$$

2.2. Evaluation Indicators. In this section, we mainly introduce the indicators of IDS.

(1) F1 Score. Assuming that we classify a sample dataset as both normal and abnormal, there are four cases of classification. As shown in Table 1, that is, the True Positive, False Positive, False Negative, and True Negative. True means that the classification is correct while False means that the classification is wrong. Positive means that the classifier is divided into normal (positive samples) and Negative means that the classifier is divided into abnormal (negative samples):

(1) True Positive: normal instance is detected correctly.

(2) False Positive: abnormal instance is incorrectly classified as normal.

(3) False Negative: normal instance is misclassified as abnormal one.

(4) True Negative: abnormal instance is detected correctly.

Table 1: Detection results.

	Relevant	Not relevant
Detected	True positives (TP)	False positives (FP)
Not detected	False negative (FN)	True negatives (TN)

Precision P represents the proportion of relevant instances among the detected instances. P can be obtained by the following formula:

$$P = \frac{\text{TP}}{\text{TP} + \text{FP}}. \tag{9}$$

Recall that R represents the proportion of relevant instances that have been detected over the total amount of relevant instances. R can be obtained by the following formula:

$$R = \frac{\text{TP}}{\text{TP} + \text{FN}}. \tag{10}$$

Actually, indicators of P and R are sometimes contradictory, and thus $F1$ score is the common evaluation indicator. $F1$ score is the weighted average of P and R which can be obtained by the following formula:

$$F1 \text{ Score} = \frac{(\alpha^2 + 1) P * R}{\alpha^2 (P + R)}. \tag{11}$$

More especially, where $\alpha = 1$, $F1$ score will get the new formula, and thus

$$F1 \text{ Score} = \frac{2PR}{P + R}. \tag{12}$$

(2) The Calculation Time. The calculation time t of the IDS detection algorithm. t contains the mode construction time and the detection time of proposed method.

3. A New IDS System for Fog Computing

In this section, a new IDS system for fog computing is presented. The main steps of this system are shown as follows. As shown in Figure 2, our proposed IDS system mainly consists of three steps: Step 1: the data preprocess; Step 2: data normalization; Step 3: decision tree detection. And the main work of each step is shown as follows.

Step 1 (data preprocess). The given dataset is usually composed of numbers and strings. We cannot compare the value of string directly, and thus we need to digitize the string by using string replace operation. The details are shown in Algorithm 1. We firstly traverse the given dataset D and find all the strings S in dataset D and obtain the corresponding columns c by using find () function (Line (1) to Line (3)). Secondly, we call the replace function to replace S with random number m (Line (7) to (9)). Finally the processed dataset D' is retuned. In addition, D' will be the input for Step 2.

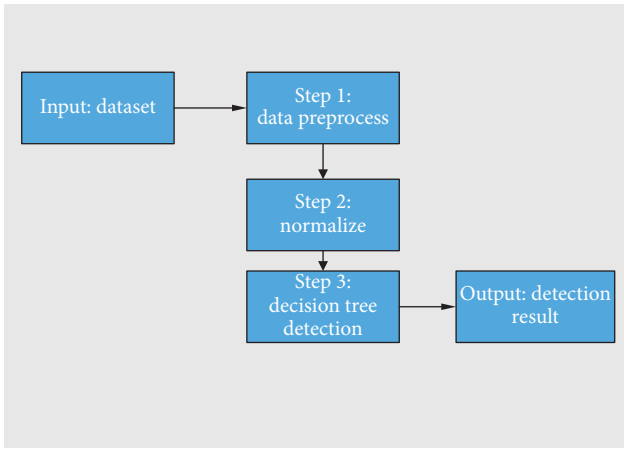

Figure 2: A new IDS detection system for fog computing.

Input: Dataset D
Output: processed Dataset D'
(1) **for** $n = 1$ to N do
(2) **for** $m = 1$ to N
(3) $(S, c) \leftarrow \text{find}(D)$
(4) **end for**
(5) **end for**
(6) **for** $n = 1$ to c do
(7) $D'(S) \leftarrow D(m)$
(8) **end for**
(9) **return** D'

Algorithm 1: String replace method.

Step 2 (data normalization). Notice that the range of numbers in D' may not uniform. That means large numbers of columns will cause the role of small columns to be ignored, and in fact there are some small numbers of columns that may play a very important role. And thus we should perform the normalization process before executing the detection algorithm; the object of normalization is to make the characteristic data shrink [0-1]. The main content is shown in Algorithm 2. Firstly, we randomly select $m\%$ of the training dataset D_1 from D' as the training dataset D_1 and the remaining D_2 equals $(1 - m\%)$ as the testing dataset (Line (1)). Then we obtain the normalization results X_train and X_test by using normalization function (Line (2) to Line (3)). Obviously, X_train and X_test will be the input for Algorithm 3.

Step 3 (decision tree detection method). In this step, we mainly construct the decision tree by using given training dataset X_train and then get the detection result of test dataset X_test. As shown in Algorithm 3, firstly, the decision tree mode mo is established by using CART function according to the related formula (2) to (8) illustrated in Section 2.1 (Line 1). Secondly, the labels in X_test are obtained by using the mode mo (Line 2). Last but not least, we obtain the results of $F1$ score and calculation time t by using statistics () function

Input: Dataset D'
Output: processed Dataset X_train, X_test
(1) $D_1, D_2 \leftarrow \text{train\_test\_split}(D', m\%)$
(2) $X\_train \leftarrow \text{normalization}(D_1)$
(3) $X\_test \leftarrow \text{normalization}(D_2)$
(4) **return** X_train, X_test

Algorithm 2: Data normalization method.

Input: X_train, X_test
Output: $F1$ Score, calculation time t
(1) $\text{mo} \leftarrow \text{CART}(X\_train)$
(2) $Y2 \leftarrow \text{mo}(X\_test)$
(3) $(F1\ \text{Score}, t) \leftarrow \text{statistics}()$
(4) **return** $F1$ Score, t

Algorithm 3: Decision tree detection method.

according to the related formula (9) to (12) illustrated in Section 2.2 (Line 3).

4. Experimental Evaluation

4.1. Experimental Environment. In this section, we evaluate our proposed IDS system on KDDCUP99 dataset. The experiment is implemented by Python on a windows 10 Operating System, where the processor is Inter Core i7 2.7 GHZ, the RAM is 16 GB, and the main software platform is Eclipse and Anaconda 2.7 SCIkit-learn.

4.2. The Introduce of Dataset. For research of IDS, a large number of valid experimental data is needed. Data collection can be obtained through some capture tools, such as TCPdump, Libdump, and Wireshark, and then connection record is generated as the data source for IDS. In this study, we use KDDCUP99 [27] dataset for our test. The dataset is a 9-week network connection data collected from a simulated LAN of US Air Force. The dataset contains two kinds, the former one is 10% dataset named as KDDcup.Data.10.percent.correceted and the latter one is the full dataset named kddcup.data.corrected. Each connection record in KDDCUP99 dataset contains forty-one fixed feature attributes and a class label. Among the forty-one features, nine features are symbolic while the other ones are continuous. As shown in Table 2, the class identifier indicates that the connection record is normal or a specific kind of attack. In addition, we can see that the DOS, Probing, R2L, and U2R have a more detailed division. In this study, the attack kinds are marked as numbers. The corresponding marks are shown in Tables 3 and 4. On the one hand, we will complete the detection of four kinds of attacks; on the other hand, we will complete the detection of twenty-two kinds of attacks. Meanwhile, both the 10% dataset and the full dataset are tested in our experiment. And thus we will perform four group experiments for each method: (1) four kinds of attacks

Table 2: Class identifier of KDDCUP99.

Classification	Meaning	Subclass
Normal	Normal record	normal
DOS	Denial of service attack	Back land Neptune pod smurf teardrop
Probing	Monitoring and other detection activities	Ipsweep nmap portsweep satan
R2L	Illegal access from remote machines	ftp_write guess_passwd imap multihop phf spy warezclient arezmaster
U2R	Unauthorized access of ordinary users' to privileges of administrator	buffer_overflow loadmodule perl rootkit

Table 3: Four kinds of attack classification are marked.

Normal = 0	normal = 0
DOS = 1	back = 1 land = 1 neptune = 1 pod = 1 smurf = 1 teardrop = 1
Probing = 2	ipsweep = 2 nmap = 2 portsweep = 2 satan = 2
R2L = 3	ftp_write = 3 guess_passwd = 3 imap = 3 multihop = 3 phf = 3 spy = 3 warezclient = 3 warezmaster = 3
U2R = 4	buffer_overflow = 4 loadmodule = 4 perl = 4 rootkit = 4

over 10% dataset (2); twenty-two kinds of attacks over 10% dataset; (3) four kinds of attacks over full dataset; (4) twenty-two kinds of attacks over full dataset. Notice that the 10% dataset contains all twenty-two attacks; the full dataset does not contain attacks of No. 17, No. 18, and No. 20.

4.3. Experiment Result and Discussion. We compare the experiment results from the aspects of $F1$ Score and calculation time. In order to cover all the attack kinds and ensure the effectiveness of the test results, we randomly divided the dataset, 60% of which was used as a training dataset and 40% as a test dataset. As a result, the Naïve Bayesian contains three models: MultinomialNB, BernoulliNB, and GaussianNB [32]. And therefore, we firstly test Bayesian method and find the best one for IDS. And then compare it with the other two methods. For each method, we conduct 10 group experiments and then compare their average.

4.3.1. Experiment Result Contrast of Three Modes of Bayesian. Firstly, we test the Bayesian method. The calculation time contrast results are shown in Figure 3. MultinomialNB gets the least calculation time among all the test cases, followed by BernoulliNB, and GaussianNB is the last one. And then we compare the results according to the test results of $F1$ score. Our principle is shown as follows. We firstly see the detection precision of normal class, as in the actual situation, the proportion of normal class is relatively large, and then

Table 4: Twenty-two kinds of attack classification are marked.

normal = 0
buffer_overflow = 1
pod = 2
teardrop = 3
guess_passwd = 4
portsweep = 5
ipsweep = 6
land = 7
back = 8
neptune = 9
smurf = 10
teard = 11
satan = 12
ftp_write = 13
imap = 14
multihop = 15
phf = 16
spy = 17
warezclient = 18
warezmaster = 19
loadmodule = 20
perl = 21
rootkit = 22

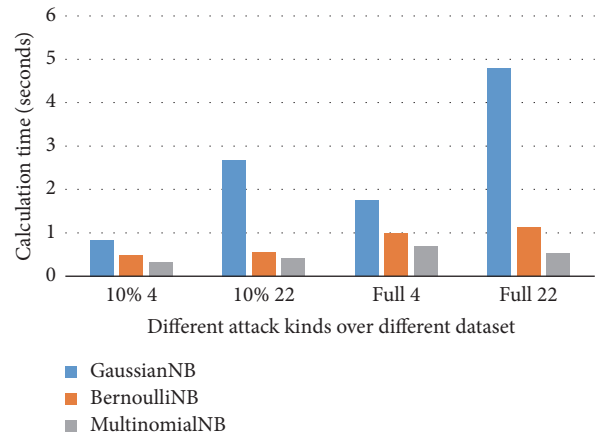

Figure 3: The calculation time contrast of three modes of Bayesian.

see the detection coverage of all attacks. As the attack type is divided into four kinds and twenty-two kinds, so we firstly discuss the detection result of three methods on four kinds of attacks and then discuss the detection result on twenty-two kinds of attacks.

(1) As shown in Figure 4, for 10% dataset, the detection precision on GaussianNB for the normal type is significantly lower than the other two methods. The BernoulliNB method is slightly lower than the MultinomialNB method for the normal type detection. As shown in Figure 5, for full dataset, detection $F1$ Score based on GaussianNB for the normal type has increased, but still lower than the other two. In

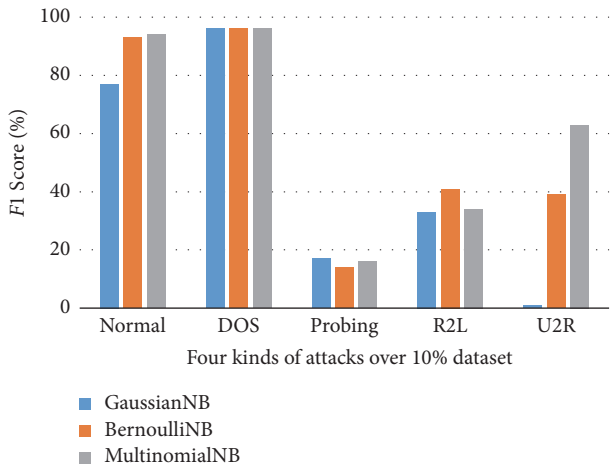

Figure 4: $F1$ score contrast of four kinds of attacks over 10% dataset of three modes.

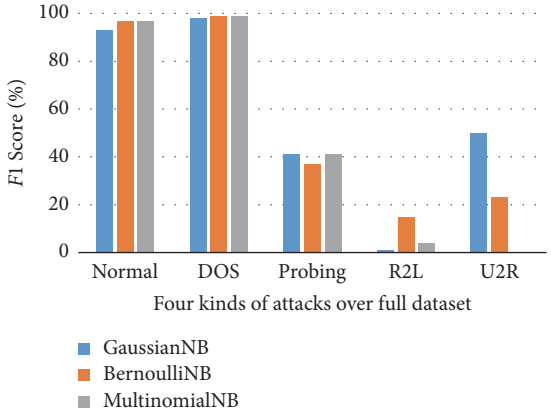

Figure 5: $F1$ score contrast of four kinds of attacks over full dataset of three modes.

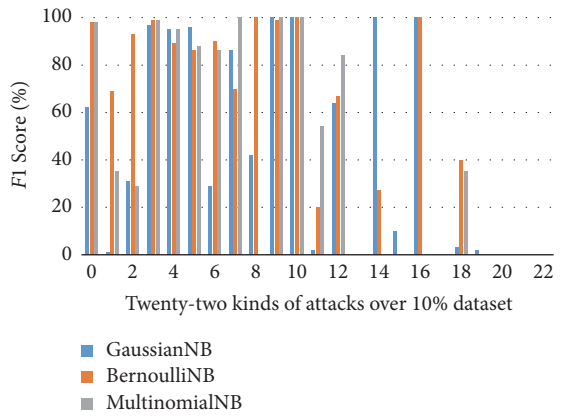

Figure 6: $F1$ score contrast of twenty-two kinds of attacks over 10% dataset of three modes.

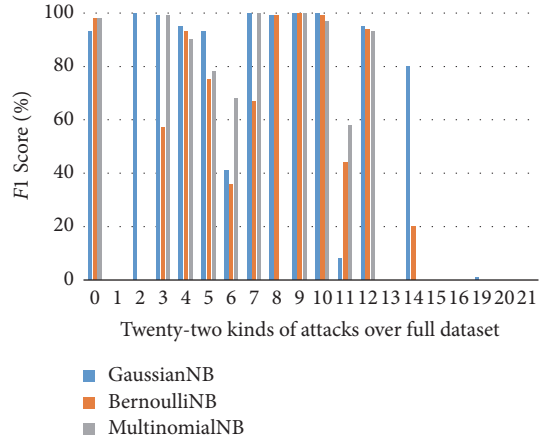

Figure 7: $F1$ score contrast of twenty-two kinds of attacks over full dataset of three modes.

addition, GaussianNB and BernoulliNB can do the detection type coverage. $F1$ Score of U2R based on MultinomialNB is 0%. However BernoulliNB is relatively stable. Although the $F1$ Score of U2R detection by GaussianNB is better than BernoulliNB, the detection $F1$ Score of R2L is much lower than BernoulliNB's, meanwhile, considering $F1$ Score on normal type by GaussianNB is lower than BernoulliNB method. In addition, the calculation time of the former one is much longer than the latter one. And thus, the BernoulliNB method is the best method for IDS.

(2) Next, we discuss the results of the three modes for detecting twenty-two attacks over both datasets. Similarly, we first discuss the normal class of test results. As shown in Figure 6, the same as a result in the above scenario, for 10% dataset, the detection $F1$ Score based on GaussianNB for the normal type is significantly lower than the other two methods. The BernoulliNB method obtains the same precision for normal type detection with the MultinomialNB method. In addition, in view of the detection $F1$ Score of twenty-two kinds of attacks, the BernoulliNB method is the best. As shown in Figure 7, for the full dataset test, the $F1$ Score of the normal class by GaussianNB has improved but is still lower than the other two methods. The analytical method is the same as above. Considering the $F1$ Score of detection for other attacks, the Bernoulli method is the best.

And thus, among the three modes of Bayesian, the BernoulliNB model is the most suitable one for IDS. Next, we will compare it with the other two methods in the next experiment.

4.3.2. Experiment Results Contrast of Three Methods. Next, we will compare BernoulliNB with decision tree and KNN. The calculation time contrast results are shown in Table 5. BernoulliNB gets the least calculation time among all the test cases, followed by decision tree, and KNN is the last one.

As shown in the Table 5, the calculation time of KNN cannot be accepted. Since KNN has the worst performance in time, there is no need for multiple experiments. The $F1$ Score results of the KNN experiment corresponding to the time in the Table 5 are shown in Table 6. KNN method over full dataset is not very good and cannot detect number 4 (U2R) attack.

Table 5: The calculation time contrast for three methods.

Method	Group			
	4 kinds 10%	22 kinds 10%	4 kinds full	22 kinds full
Decision tree	1.455921545 s	1.173570469 s	3.319417967 s	3.143889363 s
BernoulliNB	0.472016006 s	0.552608763 s	0.986687026 s	1.116946133 s
KNN	1962.25 s	/	7372.6833 s	/

Table 6: $F1$ Score of KNN method.

Dataset	Number				
	0	1	2	3	4
10% dataset	100%	100%	99%	94%	52%
Full dataset	100%	100%	100%	85%	0%

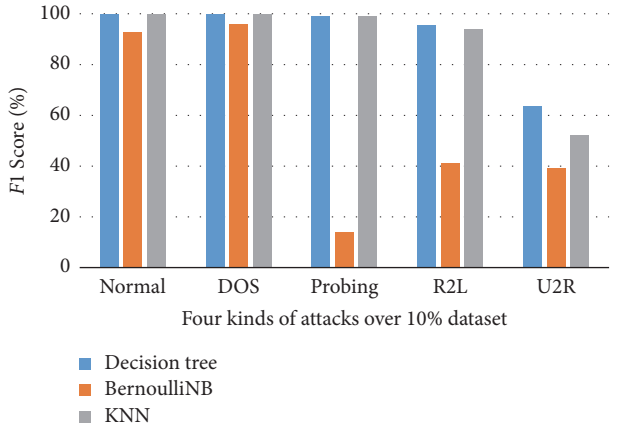

Figure 8: $F1$ score contrast of four kinds of attacks over 10% dataset of three methods.

(1) We first discuss detection results of the two methods of four kinds of attacks and then discuss the situation of twenty-two kinds of attacks. As shown in Figure 8, for each 10% dataset, the $F1$ Score of all attacks based on the decision tree is higher than the BernoulliNB method; as shown in Figure 9, for full dataset detection, all attack detection $F1$ Score on decision tree is higher than BernoulliNB except U2R.

(2) Next, we discuss the results of twenty-two attacks of the two methods over both datasets. Similarly, we firstly discuss results of the normal class. As shown in Figure 10, the decision tree obtains the same precision with BernoulliNB on No. 8 attack, No. 10 attack, and No. 16 attack. The precision of other attacks on decision tree is much higher than BernoulliNB. In particular, BernoulliNB cannot detect No. 13 attack, No. 15 attack, No. 19 attack, No. 21 attack, and No. 22 attack while decision tree methods can do it. As shown in Figure 11, for full dataset, the decision tree method obtains the same $F1$ Score with BernoulliNB on No. 9 attack. In addition, the $F1$ Score of BernoulliNB method is slightly lower than BernoulliNB for No. 4 attack. Moreover, the decision tree method is better than BernoulliNB in all other cases. In addition, the calculation time of the former one is much longer than the latter one. Above all, decision tree

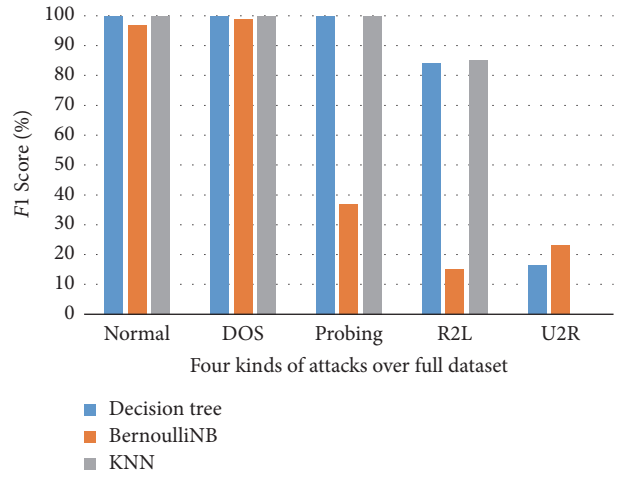

Figure 9: $F1$ score contrast of four kinds of attacks over full dataset of three methods.

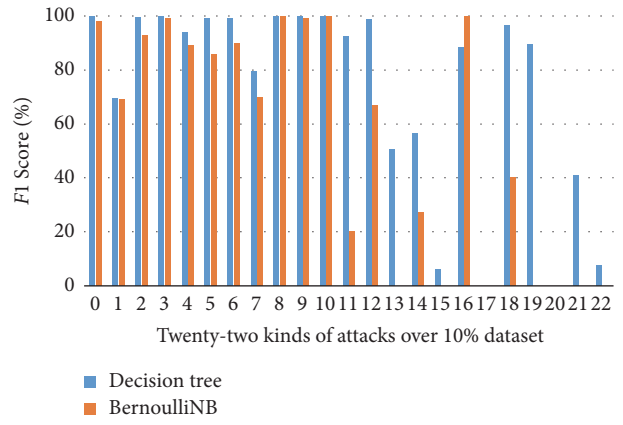

Figure 10: $F1$ score contrast of twenty-two kinds of attacks over 10% dataset.

method is the most suitable one for IDS over big data in fog environment.

4.3.3. Discussion and Performance Analysis. Among the three models of the Bayesian method, BernoulliNB tends to be the best one from the perspective of $F1$ Score. The overall $F1$ Score of the decision tree is the best. From the perspective of the calculation time, the BernoulliNB is the best. The KNN is not suitable for the large dimension although the precision is very high over 10% dataset. For IDS issue, if only the precision is considered, the decision tree algorithm is the best choice; if only the calculation time is considered, the BernoulliNB

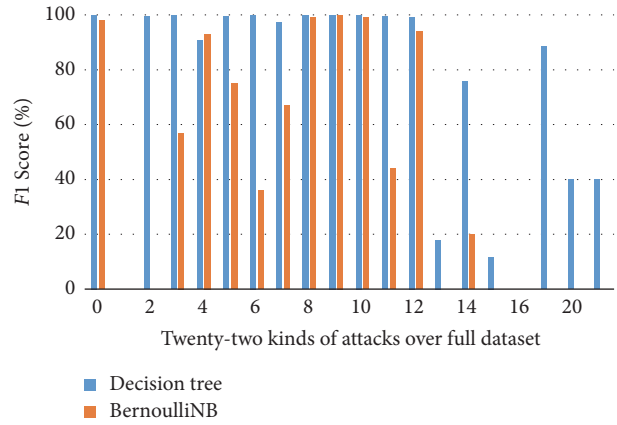

Figure 11: $F1$ score contrast of twenty-two kinds of attacks over full dataset.

algorithm is much better. From the point of view of the calculation time, although is not the best, the calculation time of the decision tree is acceptable. The authors in [24] point out that the calculation time of Naïve Bayesian is generally 7 times faster than that of decision trees by using C4.5. However, in this study, we can conclude that the decision tree based on CART is much faster. The multiple comparison of calculation time is shown as Table 7.

(1) BernoulliNB is 2.364 times faster than decision tree in the case of four kinds of attacks over full dataset. In particular, the time gap is narrowed over the situation of twenty-two kinds of attacks.

(2) In order to make the comparison more comprehensive, we simply look at it with the other two Bayesian cases. Compared with GaussianNB, the decision tree is much faster than GaussianNB over the situation of twenty-two kinds of attacks, even when compared with BernoulliNB which is the most time efficiency mode of Naïve Bayesian, MultinomialNB is only 4.857 times faster than decision tree in the worst situation.

However, taking into account the detection accuracy, as well as the coverage of the attacks, there is no doubt that the decision tree is the best choice for IDS over big data in fog computing.

Above all, our proposed IDS system is efficient and precise. As shown in Figure 1, our proposed system can be deployed in a common node of fog layer without extra requirement. According to the above experiment, we can conclude that the system performance is stable and performs very well in big data environment.

5. Related Work

Fog computing was for the first time proposed by Cisco in 2012 and defined as a highly virtualized computing platform for migrating the tasks of Cloud to network mobile users. The fog computing [4] introduces the middle layer between the cloud and the mobile users, extending the cloud-based network structure, and provides computing, storage, as well as network service between mobile devices and Cloud. The fog computing reduces unnecessary multiple communication between the cloud computing center and the mobile users [8]. It not only reduces the network delay for mobile users but also significantly reduces the link bandwidth backbone [9, 10]. Although there are many advantages of fog computing, some security issues still need to be solved. More specifically, fog computing nodes are usually composed of weak computing power. Traditional network attacks become more common in fog computing environment, such as eavesdrop or hijack the mobile user data and even attempt to destroy the fog system. Fortunately, Intrusion Detection Systems (IDS) can also be applied in fog environment [11].

After decades of development, IDS has become a more successful security technology. IDS which represented by Snort [33] has made an outstanding contribution to network security in recent years. ISS RealSecure is also well known, and it mainly consists of two parts, the engine part and the console part. The former one is responsible for detecting information and generate alarms and the latter one receives the alarm and is a central point for configuring and generating the database report. Pattern matching algorithm is one of the core technologies of IDS products. Misuse detection based on AC, BM, MWM, and other matching algorithms [15] can make IDS have a passive detection of known attacks with wide and obvious characteristics. However, modern attacks are increasingly inclined to form an unknown intrusion technology by integrating a variety of known intrusion technologies. Meanwhile, improved IDS methods usually take proactive protection based on deviation detection and user behavior anomaly detection. Statistical model, Bayesian reasoning, cluster analysis [16], and other excellent algorithms like DB can make up for the lack of pattern matching. KNN algorithm known as k nearest neighbor algorithm [17] is widely used in pattern recognition, classification, and regression [18]. Same as KNN, vector automatic classification algorithms, support vector machine [19, 20], neural network algorithm [21], Bayesian algorithm [22–24], and K means algorithm are also widely used for IDS [25, 26].

Although the IDS in tradition network has been well investigated, unfortunately directly use them in fog computing environment may not inappropriate. More specifically, the existing researches mainly present the experiments on 10% KDDCUP99 dataset. Although these methods have achieved good results, we cannot judge their efficiency when they are presented in big data environment, even in the full dataset of KDDCUP99. In addition, there are four kinds of attacks classification, as well as twenty-two attacks classification in KDDCUP99. However, the existing researchers mainly focus on the detection of four attacks but fail to consider the detection of twenty-two attacks. In order to address the aforementioned problem, we propose an IDS system based on Anaconda, we use decision tree for our IDS detection, and multimethods are compared. Although the author in [24] also uses Bayesian and decision tree methods for IDS. Different from them, we conducted a more adequate experiment. And we compare decision tree with three modes of Naïve Bayesian method, as well as KNN method. More specifically, both the 10% dataset and the full dataset are tested in our IDS system. We not only complete the detection of

Table 7: The multiple comparison of calculation time.

Method	Group			
	4 kinds 10%	22 kinds 10%	4 kinds full	22 kinds full
Decision Tree|GaussianNB	1.757	0.438	1.894	0.655
Decision Tree|BernoulliNB	3.084	2.124	3.364	2.815
Decision Tree|MultinomialNB	4.393	2.882	4.837	5.857

four kinds of attacks but also accomplish the detection of twenty-two kinds of attacks. In addition, the calculation time of each method is compared. The authors in [20] also consider the calculation time of their algorithm; however, they also only present their experiments on 10% dataset, and thus we cannot judge the performance of the algorithm over big data environment. Above all, the experiment results show that our proposed system is effective and precise.

6. Conclusion

Tradition network attacks are widely present in fog computing environment. Although the IDS in tradition network have been well investigated, unfortunately directly use of them in fog computing environment may not inappropriate. In this study, we propose a system based on the decision tree, multimethods are compared with this one, not only the 10% dataset but also the full dataset is tested, and the experiment results show that our system is effective. In addition, we also compared the detection time for each method. In the case of guaranteed accuracy, although the decision tree time is not the best one, the calculation time is also acceptable. Above all, our IDS system can be used in fog computing environment over big data. In our future, we will engage in the research of the IDS for other kinds of attacks.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work is supported by Quanzhou Science and Technology Project (no. 2015Z115), the Scientific Research Foundation of Huaqiao University (no. 14BS316), the Education and Scientific Research Projects of Young and Middle-aged Teachers in Fujian Province (JZ160084), College Students Innovation and Entrepreneurship Project (201710385023), and China Scholarship Council award to Dr. Kai Peng for one year's research abroad at the University of British Columbia.

References

[1] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, "Fog computing and its role in the internet of things," in *Proceedings of the 1st ACM Mobile Cloud Computing Workshop, MCC '12*, pp. 13–15, ACM, Helsinki, Finland, August 2012.

[2] L. M. Vaquero and L. Rodero-Merino, "Finding your way in the fog: towards a comprehensive definition of fog computing," *ACM SIGCOMM Computer Communication Review Archive*, vol. 44, no. 5, pp. 27–32, 2014.

[3] X. Xu, X. Zhang, M. Khan, W. Dou, S. Xue, and S. Yu, "A balanced virtual machine scheduling method for energy-performance trade-offs in cyber-physical cloud systems," *Future Generation Computer Systems*, 2017.

[4] T. H. Luan, L. Gao, Z. Li, Y. Xiang, G. Wei, and L. Sun, "Fog computing: focusing on mobile users at the edge," 2015, https://arxiv.org/abs/1502.01815.

[5] G. I. Klas, Fog computing and mobile edge cloud gain momentum open fog consortium, etsi mec and cloudlets, 2015.

[6] X. L. Xu, X. Zhao, F. Ruan et al., "Data placement for privacy-aware applications over big data in hybrid clouds," *Security and Communication Networks*, vol. 2017, Article ID 2376484, 15 pages, 2017.

[7] I. Stojmenovic, S. Wen, X. Huang, and H. Luan, "An overview of Fog computing and its security issues," *Concurrency Computation*, vol. 28, no. 10, pp. 2991–3005, 2016.

[8] G. M. Relations, "Greyhound launches "Blue ?" An exclusive WiFi enabled onboard entertainment system," https://www.greyhound.com/en/about/media/2013/07-08-2013.

[9] W. S. Shi, H. Sun, J. Cao, Q. Zhang, and W. Liu, "Edge computingan emerging computing model for the internet of everything," *Journal of Computer Research and Development*, vol. 54, no. 5, 2017.

[10] Y. Fan and Q. Zhu, "Cloud/Fog computing system architecture and key technologies for south-north water transfer project safety," *Wireless Communications and Mobile Computing*, 2017.

[11] C. Modi, D. Patel, B. Borisaniya, H. Patel, A. Patel, and M. Rajarajan, "A survey of intrusion detection techniques in cloud," *Journal of Network and Computer Applications*, vol. 36, no. 1, pp. 42–57, 2013.

[12] A. Milenkoski, M. Vieira, S. Kounev, A. Avritzer, and B. D. Payne, "Evaluating computer intrusion detection systems: a survey of common practices," *ACM Computing Surveys*, vol. 48, no. 1, pp. 1–41, 2015.

[13] J. P. Anderson, "Computer security threat monitoring and surveillance," 1980.

[14] D. E. Denning, "An intrusion-detection model," *IEEE Transactions on Software Engineering*, vol. SE-13, no. 2, pp. 222–232, 1987.

[15] Y. Li, H. Li, X. Qian, and Y. Zhu, "A review and analysis of outlier detection algorithms," *Analysis of Outlier Detection Algorithms*, vol. 6, p. 5, 2002.

[16] T. Hastie and R. Tibshirani, "Discriminant adaptive nearest neighbor classification," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 18, no. 6, pp. 607–616, 1996.

[17] J. Lu, Z. Wu, Y. Wang, and Y. Lu, "Research on abnormal behavior detection based on kNN algorithm," *Computer Engineering*, vol. 7, p. 48, 2007.

[18] K. Li, J. Zhao, H. Hong, S. Tian, and J. Zhao, "One-class support vector machine model for intrusion detection," *China Safety Science Journal*, vol. 13, no. 6, pp. 72–76, 2003.

[19] H. R. Zhang and Z. Z. Han, "An improved sequential minimal optimization learning algorithm for regression support vector machine," *Journal of Software. Ruanjian Xuebao*, vol. 14, no. 12, pp. 2006–2013, 2003.

[20] W. Zhang and J. Fan, "Cloud architecture intrusion detection system based on KKT condition and hyper-sphere incremental SVM algorithm," *Journal of Computer Applications*, vol. 35, no. 10, pp. 2886–2890, 2015.

[21] T. H. Dai, "Intrusive detection based on genetic neural networks," *China Safety Science Journal*, vol. 16, no. 2, pp. 103–108, 2006.

[22] X. L. Shao, Y. W. Liu, M. J. Geng, and J. B. Han, "The parallel Implementation of MapReduce for the Bayesian Algorithm to detect botnets," *CAAI Transactions on Intelligent System*, vol. 1, pp. 26–33, 2014.

[23] S. Wang, H. Zou, Q. Sun, and F. Yang, "Bayesian approach with maximum entropy principle for trusted quality of web service metric in e-commerce applications," *Security and Communication Networks*, vol. 5, no. 10, pp. 1112–1120, 2012.

[24] N. B. Amor, S. Benferhat, and Z. Elouedi, "Naive Bayes vs decision trees in intrusion detection systems," in *Proceedings of the 2004 ACM symposium on applied computing (SAC '04)*, pp. 420–424, March 2004.

[25] H. C. Liu, X. N. Hou, and Z. Yang, "Design of intrusion detection system based on improved K-means algorithm," *Computer Technology and Development*, vol. 1, pp. 101–105, 2016.

[26] W. L. Al-Yaseen, Z. A. Othman, and M. Z. A. Nazri, "Multi-level hybrid support vector machine and extreme learning machine based on modified K-means for intrusion detection system," *Expert Systems with Applications*, vol. 67, pp. 296–303, 2017.

[27] http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html.

[28] Scikit-learn, http://scikit-learn.org/stable/index.html.

[29] J. R. Quinlan, *C4.5, Programs for machine learning*, Morgan Kaufmann, San Mateo Ca, 1993.

[30] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone, *Classification and Regression Trees*, Wadsworth and Brooks, California, Calif, USA, 1984.

[31] H. Li. Statistical learning, *method*, Tsinghua University Press, Beijing, 2012.

[32] C. D. Manning, P. Raghavan, and H. Schuetze, *Introduction to Information Retrieval*, Cambridge University Press, Cambridge, UK, 2008.

[33] T. Cover and P. Hart, "Nearest neighbor pattern classification," *IEEE Transactions on Information Theory*, vol. 13, no. 1, pp. 21–27, 1967.