Hindawi
Journal of Electrical and Computer Engineering
Volume 2017, Article ID 4975343, 9 pages
https://doi.org/10.1155/2017/4975343

Research Article

A DDoS Attack Detection Method Based on Hybrid Heterogeneous Multiclassifier Ensemble Learning

Bin Jia,[1] Xiaohong Huang,[1] Rujun Liu,[2] and Yan Ma[1]

[1]*Information and Network Center, Institute of Network Technology, Beijing University of Posts and Telecommunications, Beijing 100876, China*
[2]*School of CyberSpace Security, Beijing University of Posts and Telecommunications, Beijing 100876, China*

Correspondence should be addressed to Bin Jia; jb_qd2010@bupt.edu.cn

Received 22 October 2016; Accepted 11 January 2017; Published 15 March 2017

Academic Editor: Jun Bi

The explosive growth of network traffic and its multitype on Internet have brought new and severe challenges to DDoS attack detection. To get the higher True Negative Rate (TNR), accuracy, and precision and to guarantee the robustness, stability, and universality of detection system, in this paper, we propose a DDoS attack detection method based on hybrid heterogeneous multiclassifier ensemble learning and design a heuristic detection algorithm based on Singular Value Decomposition (SVD) to construct our detection system. Experimental results show that our detection method is excellent in TNR, accuracy, and precision. Therefore, our algorithm has good detective performance for DDoS attack. Through the comparisons with Random Forest, *k*-Nearest Neighbor (*k*-NN), and Bagging comprising the component classifiers when the three algorithms are used alone by SVD and by un-SVD, it is shown that our model is superior to the state-of-the-art attack detection techniques in system generalization ability, detection stability, and overall detection performance.

1. Introduction

The explosive growth of network traffic and its multitype on Internet have brought new and severe challenges to network attack behavior detection. Some traditional detection methods and techniques have not met the needs of efficient and exact detection for the diversity and complexity of attack traffic in the high-speed network environment, especially such as DDoS attack.

Distributed Denial of Service (DDoS) attack is launched by some remote-controlled Zombies. It is implemented by forcing a kidnapped computer or consuming its resources, such as CPU cycle, memory, and network bandwidth. Moreover, Palmieri et al. [1] described the attack that exploits computing resources to waste energy and to increase costs. With the network migrating to cloud computing environments, the rate of DDoS attacks is growing substantially [2]. For DDoS attack detection, the previous researches in recent years mainly include the following.

In 2014, Luo et al. [3] developed a mathematical model to estimate the combined impact of DDoS attack pattern and network environment on attack effect by originally capturing the adjustment behaviors of victim TCPs congestion window. In 2015, Xiao et al. [4] presented an effective detection approach based on CKNN (*K*-nearest neighbor's traffic classification with correlation analysis) and a grid-based and named r-polling method to detect DDoS attack. The methods can reduce training data involved in the calculation and can exploit correlation information of training data to improve the classification accuracy and to reduce the overhead caused by the density of training data. In 2016, Saied et al. [5] designed a model to detect and mitigate the known and unknown DDoS attack in real-time environments. They chose an Artificial Neural Network (ANN) algorithm to detect DDoS attack based on specific characteristic features that separate DDoS attack traffic from normal traffic.

However, the existing detection methods still suffer from low True Negative Rate (TNR), accuracy, and precision. And

their methods or models are homogeneous, so the robustness, stability, and universality are difficult to be guaranteed. To address the abovementioned problems, in this paper, we propose the DDoS attack detection method based on hybrid heterogeneous multiclassifier ensemble learning.

Ensemble learning finishes the learning task by structuring and combining multiple individual classifiers. It is homogeneous for the ensemble of the same type of individual classifiers, and this kind of individual classifier is known as "base classifier" or "weak classifier." Ensemble learning can also contain the different types of individual classifiers, and the ensemble is heterogeneous. In heterogeneous ensemble, the individual classifiers are generated by different learning algorithms. The classifiers are called as "component classifier." For the research of homogeneous base classifier, there is a key hypothesis that the errors of base classifier are independent of each other. However, for the actual attack traffic detection, they apparently are impossible. In addition, the accuracy and the diversity of individual classifiers conflict in nature. When the accuracy is very high, increasing the diversity becomes extremely difficult. Therefore, to generate the robust generalization ability, the individual classifiers ought to be excellent and different.

An overwhelming majority of classifier ensembles are currently constructed based on the homogeneous base classifier model. It was proved to obtain a relatively good classification performance. However, according to some theoretical analyses, while error correlation between every two individual classifiers is smaller, the error of ensemble system is smaller. Simultaneously, while the error of classifier is increased, the negative effect came into being. Therefore, the ensemble learning model for homogeneous individual classifiers cannot satisfy the needs of the higher ensemble performance [6].

According to the different measured standard in [7], while the value of diversity is larger, the difference is larger, and the classifying precision of classifier is higher. Through the abovementioned analysis, the heterogeneous multiclassifier ensemble learning owns more remarkable and more predominant generalization performance than the homogeneous multiclassifier ensemble learning.

This paper makes the following contributions. (i) To the best of our knowledge, this is the first attempt to apply the heterogeneous multiclassifier ensemble model to DDoS attack detection, and we provide the system model and its formulation. (ii) We design a heuristic detection algorithm based on Singular Value Decomposition (SVD) to construct the heterogeneous DDoS detection system, and we conduct thorough numerical comparisons between our method and several famous machine learning algorithms by SVD and by un-SVD.

The rest of this paper is organized as follows. Section 2 describes the system model and the related problem formulation. Section 3 presents DDoS detection algorithm in heterogeneous classification ensemble learning model. Our experimental results and analysis are discussed in Section 4. In Section 5, we conclude this paper.

2. System Model and Problem Formulation

2.1. System Model. The classification learning model based on Rotation Forest and SVD aims at building the accurate and diverse individual component classifiers. Here, the model is used for DDoS attack detection. Rodríguez et al. [8] applied Rotation Forest and Principal Component Analysis (PCA) algorithm to extract features subsets and to reconstruct a full feature set for each component classifier in the ensemble model. However, PCA has indeed some weaknesses. For instance, PCA sees all samples as a whole and searches an optimal linear map projection on the premise of minimum mean-square error. Nonetheless, categorical attributes are ignored. The ignored projection direction is likely to include some important divisible information. In order to overcome the abovementioned shortcomings, in this paper, we use SVD to substitute for PCA.

There are two key points to construct heterogeneous multiclassifier ensemble learning model [9]. The first one is the selection of individual classifiers. The second one is the assembly of individual classifiers. We discuss the two issues as follows.

(i) Firstly, classifiers in an ensemble should be different from each other; otherwise, there is no gain in combining them. These differences cover diversity, orthogonality, and complementarity [10]. Secondly, Kuncheva and Rodríguez [11] proposed static classifier selection and dynamic classifier selection. The difference of the two methods is whether or not evaluation of competence is implemented during the classification. Because of the characteristic of prejudged competence, the dynamic classifier selection approach is better than the static classifier selection. Last but not least, the powerful generalization ability is necessary in heterogeneous classification ensemble model. Irrelevance between every two individual classifiers is desired as much as possible, and the error of every individual classifier itself ought to be smaller. The interrelation between error rate of integrated system and correlation of individual classifiers is given [6] by

$$E = \left(\frac{1 + \rho (N - 1)}{N} \right) \overline{E} + E_{\text{Optimal Bayes}}, \tag{1}$$

where E denotes Ensemble Generalization Error (EGE). $\overline{E}$ denotes the mean of these individual generalization errors. $E_{\text{Optimal Bayes}}$ denotes False Recognition Rate (FRR) that is got by using the Bayesian rules when all conditional probabilities are known. Therefore, the above formula shows that the ensemble error is smaller when individual differences are bigger and individual errors are smaller.

(ii) The majority voting method is chosen as our combined strategy of all component classifiers. For the prediction label of every record in testing data set, we choose those labels whose votes are more than half as final predicting outcomes. The majority voting is given by

$$H(x) = P_i, \quad \text{if} \sum_{i=1}^{t} h_i^j (x) > \frac{1}{2} \sum_{l=1}^{m} \sum_{i=1}^{t} h_i^l (x), \tag{2}$$

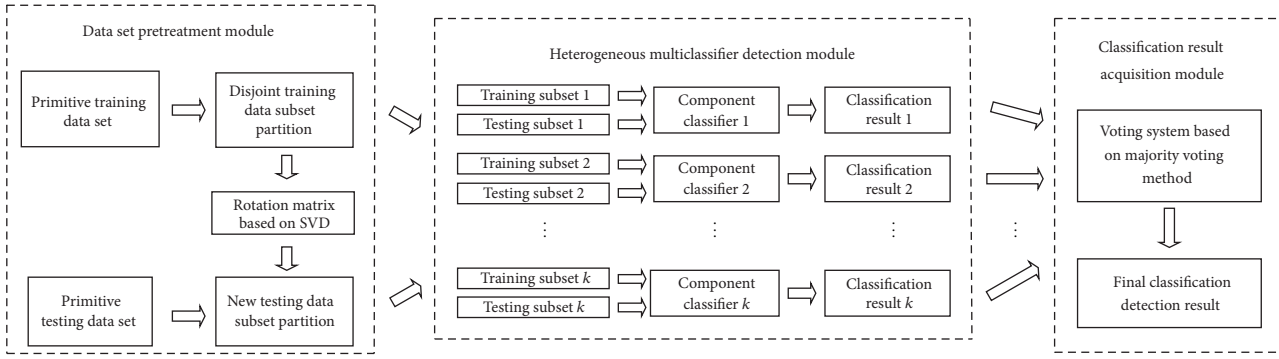


Figure 1: Hybrid heterogeneous multiclassifier ensemble classification model.

where the m-dimension vector $(h_i^1(x); h_i^2(x); \ldots; h_i^m(x))$ is denoted as prediction output. h_i is on the sample x, and $h_i^j(x)$ is the output on the class label P_i.

As shown in Figure 1, the proposed hybrid heterogeneous multiclassifier ensemble classification model includes three primary modules, and they are Data Set Pretreatment Module, Heterogeneous Multiclassifier Detection Module, and Classification Result Acquisition Module. In Data Set Pretreatment Module, the primitive training data set is split first into k disjoint training data subsets based on SVD. The new training data subset is generated by the linearly independent base transformation. Secondly, the primitive testing data sets are split also into k data subsets corresponding to the features of new training data subsets. Finally, the new testing data subsets are generated by Rotation Forest. In Heterogeneous Multiclassifier Detection Module, the k new training data subsets and testing data subsets are used as the input of k component classifiers. Next, the k classification detection results are got. In Classification Result Acquisition Module, the voting system votes on k results based on the majority voting method. Then, the final classification detection result is obtained.

2.2. Problem Formulation. We assume that the $m \times n$ matrix denotes a training data set, where m represents the number of network traffic records, and n represents the number of characteristic attributes in a record. In addition, D_i denotes the ith classifier in heterogeneous ensemble. Here, all component classifiers can be trained in parallel. The whole features of training data set are split into k disjoint data subsets. Each feature subset contains f ($f = [n/k]$) features. The SVD method is applied to every split subset.

Suppose that C is the $m \times f$ training subset matrix, U is the $m \times m$ square matrix, and V is the $f \times f$ square matrix. Here, the column of U is orthogonal eigenvector of CC^T and the column of V^T is orthogonal eigenvector of C^TC. In addition, r is rank of the matrix C. The SVD is shown by

$$C = U \sum V^T, \tag{3}$$

where $\sum$ is the $m \times f$ matrix. $\sum_{ii} = \sqrt{\lambda_i}$, and the different values of $\sum_{ii}$ are arranged by the descending order. The values on the other locations are zero.

Then, the computational formulae of the singular value and the eigenvectors are given by

$$\left(C^TC\right) v_i = \lambda_i v_i, \tag{4}$$

$$\sigma_i = \sqrt{\lambda_i}, \tag{5}$$

where v_i represents the eigenvector and σ_i represents the singular value.

The new training data subset C' whose all features are linearly independent is obtained by

$$C' = CV^T. \tag{6}$$

So far, the singular value and their eigenvectors of the corresponding eigenvalue by SVD for every subset are obtained. Next, the rotation matrix is got. It is denoted by

$$R_C = \begin{bmatrix} V_1 & [0] & \cdots & [0] \\ [0] & V_2 & \cdots & [0] \\ \vdots & \vdots & \ddots & \vdots \\ [0] & [0] & \cdots & V_k \end{bmatrix}, \tag{7}$$

where V_i ($i = 1, 2, \ldots, k$) represents the eigenvector set of C^TC for the ith training data subset C in the main diagonal position. And the null vectors are in the other positions.

The primitive testing data set C_p^T is split also into k data subsets corresponding to the features of new training data subsets. In order to get the new testing data set C^T, similarly, the linearly independent base transformation is operated by

$$C^T = C_p^T R_C. \tag{8}$$

One of the keys for good performance of ensembles is the diversity. There are several ways to inject diversity into an ensemble; the most common is the use of sampling [12]. To consider adequately the differentiation and complementarity between every different classification algorithm, in this paper, we select three typical machine learning algorithms as individual component classifiers, and they are Bagging [13], Random Forest [14], and k-Nearest Neighbor (k-NN) [15],

respectively. Firstly, Bagging is the best famous representative of parallel ensemble learning methods. It employs "Random Sampling" in sampling data set. The algorithm focuses mainly on decreasing variance. Secondly, "Random Feature Selection" is farther introduced in the training process for Random Forest. In the constructing process of individual decision tree, Bagging uses "Deterministic Decision Tree" where all features in node are considered in choosing to divide property. However, "Stochastic Decision Tree" is used in Random Forest, where an attribute subset is considered. Thirdly, k-NN classifies by measuring distance between every two different feature values. It usually employs Euclidean distance by

$$d = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2 + \cdots + (x_n - y_n)^2}, \quad (9)$$

where $(x_1, x_2, \ldots, x_n)$ and $(y_1, y_2, \ldots, y_n)$ represent two different sample records which have n features, respectively.

In addition, a statistical test should be employed to eliminate the bias in the comparison of the tested algorithms. In our model, we use the statistical normalization [16] as a statistical testing method. The method's purpose is to convert data derived from any normal distribution to standard normal distribution. And 99.9% samples of the attribute are scaled into $[-3, 3]$ in the method. The statistical normalization is defined by

$$x_i = \frac{v_i - \mu}{\sigma}, \quad (10)$$

where μ is the mean of n values for a given attribute and σ is its standard deviation. The mean and the standard deviation are denoted by

$$\mu = \frac{1}{n}\sum_{i=1}^{n} v_i, \quad (11)$$

$$\sigma = \sqrt{\frac{1}{n}\sum_{i=1}^{n} (v_i - \mu)^2}. \quad (12)$$

3. A Detection Algorithm in Heterogeneous Classification Ensemble Model

In this section, we first describe DDoS attack detection process in heterogeneous multiclassifier ensemble model, and then the detection algorithm based on SVD is presented.

Firstly, all primitive training data set and all testing data set are split into k disjoint data subsets by the same feature fields, respectively. Secondly, the new training data subsets and the new testing data subsets are got by linearly independent base transformation based on SVD and Rotation Forest method. Thirdly, every new training data subset and every new testing data subset are normalized by the statistical normalization in a batch manner, and then they are input into every component classifier to classify and learn. The k classification detection results are acquired. Finally, the voting system votes on k results based on the majority voting method and outputs the final classification detection result. Here, we propose a heuristics algorithm to keep the stronger generalization and sufficient complementarity.

The classification detection algorithm is shown as follows.

Input

C: a training data set: the $m \times n$ matrix

C_p^T: a testing data set: the $l \times n$ matrix

k: the number of data subsets and the number of component classifiers

Step 1. Split all features into k subsets: F_i $(i = 1, 2, \ldots, k)$, and each feature subset contains f $(f = [n/k])$ features.

Step 2. **For** $i = 1$ to k **do.**

Step 3. Apply SVD on the training subset: the $m \times f$ matrix.

Step 4. Get the linearly independent column eigenvector matrix V^T.

Step 5. Get the new training data subset: C' $(C' = CV^T)$.

Step 6. Get the new testing data subset: C'_T $(C'_T = C_T V^T)$.

Step 7. C' and C'_T are normalized by the statistical normalization in a batch manner.

Step 8. C' and C'_T are input into the component classifier.

Step 9. Get the classification label.

Step 10. **End for**.

Step 11. Input the results of k component classifiers into the voting system based on the majority voting method.

Output

The final label of a testing data record, label = {Normal, DDoS}

In our algorithm, in order to select the component classifiers, we use the parallelization principle. The component classifiers have no strong dependencies by the principle. In addition, we select Bagging, Random Forest, and k-NN algorithms as the component classifiers. In Bagging, the CART algorithm is used as the base learner of our heterogeneous classification ensemble model.

4. Experimental Results and Analysis

In this section, we discuss how to apply the heterogeneous classification ensemble model in detecting DDoS attack traffic. We first present the data set and the data pretreatment method used in our experiments. Then, the experimental results are given, and we analyze and make comparisons with the homogeneous models based on three selected algorithms by SVD and by un-SVD. Here, the computer environment to run our experiments is listed in Table 1.

Table 1: Computer experimental condition.

CPU	Memory	Hard disk	OS	MATLAB
Intel® Xeon® CPU E5-2640 v2 @2.00 GHz 2.00 GHz (2 processors)	32 GB	2 TB	Windows Server 2008 R2 Enterprise	R2013a (8.1.0.604) 64-bit (win64)

In this paper, we use the famous Knowledge Discovery and Data mining (KDD) Cup 1999 dataset [17, 18] as the verification of our detection model. This data set has been widely applied to research and evaluate network intrusion detection methods [19, 20]. KDD CUP 1999 data set comprises about five million network traffic records, and it provides a training subset of 10 percent of network record and a testing subset. Every record of the data set is described by four types of features, and they are TCP connection basic features (9 features), TCP connective content features (13 features), time-based network traffic statistical characteristics (9 features), and host-based network traffic statistical characteristics (10 features), respectively. All 41 features in the four types are shown in Table 2.

In addition, KDD CUP 1999 data set covers four main categories of attack, and these are DoS, R2L, U2R, and Probing. Because the traffic records of "neptune" and "smurf" for DoS account for more than 99% and 96% in the abovementioned training subset and testing subset, we choose the two types for DoS as our algorithm evaluation and comparison with the three famous existing machine learning algorithms in this paper.

4.1. Data Pretreatment. Firstly, because the training subset of 10 percent and the "corrected" testing subset in KDD CUP 1999 data set include hundreds of thousands of network records, the hardware configuration of our sever cannot load the calculation to process the abovementioned data sets. Here, we use the built-in "random ()" method in MySQL to randomly select one in every ten records of the abovementioned training subset and testing subset as our data sets. The data sets used in our experiments are listed in Table 3.

Secondly, for each network traffic record, it includes the information that has been separated into 41 features plus 1 class label [21] in KDD CUP 1999, and there are 3 nonnumeric features in all features. In our experiments, we also transform the type of three features into the numeric type. The conversion process is listed as follows:

(i) TCP, UDP, and ICMP in the "protocol_type" feature are marked as 1, 2, and 3, respectively.

(ii) The 70 kinds of "service" for the destination host are sorted by the percentage in the training subset of 10 percent. We get the top three types, and they are ecr_i, private and http. The three types account for over 90%. The ecr_i, private, http, and all other types are marked as 1, 2, 3, and 0, respectively.

(iii) The "SF" is marked as 1, and the other ten false connection statuses are marked as 0 in the "flag" feature.

Table 2: All 41 features in the four types.

Number	
	TCP connection basic features
(1)	duration
(2)	protocol_type
(3)	service
(4)	flag
(5)	src_bytes
(6)	dst_bytes
(7)	land
(8)	wrong_fragment
(9)	urgent
	TCP connective content features
(10)	hot
(11)	num_failed_logins
(12)	logged_in
(13)	num_compromised
(14)	root_shell
(15)	su_attempted
(16)	num_root
(17)	num_file_creations
(18)	num_shells
(19)	num_access_files
(20)	num_outbound_cmds
(21)	is_hot_login
(22)	is_guest_login
	Time-based network traffic statistical characteristics
(23)	count
(24)	srv_count
(25)	serror_rate
(26)	srv_serror_rate
(27)	rerror_rate
(28)	srv_rerror_rate
(29)	same_srv_rate
(30)	diff_srv_rate
(31)	srv_diff_host_rate
	Host-based network traffic statistical characteristics
(32)	dst_host_count
(33)	dst_host_srv_count
(34)	dst_host_same_srv_rate
(35)	dst_host_diff_srv_rate
(36)	dst_host_same_src_port_rate
(37)	dst_host_srv_diff_host_rate
(38)	dst_host_serror_rate
(39)	dst_host_srv_serror_rate
(40)	dst_host_rerror_rate
(41)	dst_host_srv_rerror_rate

Table 3: Data sets used in our experiments.

Category	Training data set	Testing data set
Normal	9728	6059
DoS	38800	22209

Table 4: Confusion matrix.

	Predicted DoS	Predicted normal	Total
Original DoS	TP	FN	P
Original normal	FP	TN	N
Total	P'	N'	$P + N (P' + N')$

4.2. Fivefold Cross-Validation. Cross-validation is an effective statistical technique to ensure the robustness of a model. In this paper, to improve the reliability of the experimental data and to verify the availability of our model, a fivefold cross-validation approach is used in our experiments.

The training data set is randomly split into five parts. In turn, we take out one part as the actual training data set and the other four parts and testing data set as the final testing data set. The aforementioned statistical normalization method in Section 2.2 is employed as the data statistical test.

4.3. Evaluation Index. Whether normal or attack for the network traffic belongs to the category of the binary classification, we need some evaluation indexes to evaluate it. In this paper, we use three typical indexes to measure our detection model, and they are TNR, accuracy, and precision. Here, TNR denotes the proportion of normal samples that are correctly recognized as normal samples in the testing data set. It reflects the veracity that detection model discerns normal samples. Accuracy denotes the proportion between the number of correctly classified samples and the total number of samples in the testing data set. It reflects the distinguishing ability to differentiate normal samples from attack samples. Precision denotes the proportion of true attack samples in all attack samples recognized by detection model in the testing data set. TNR, accuracy, and precision are formulated as follows:

$$\text{TNR} = \frac{\text{TN}}{\text{FP} + \text{TN}}, \tag{13}$$

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FP} + \text{TN} + \text{FN}}, \tag{14}$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}. \tag{15}$$

The performance of a classification detection model is evaluated by the counts of records for the normal samples and the attack samples. The matrix is called as the confusion matrix [22] based on the abovementioned counts. The matrix is listed in Table 4, where

(1) TP (True Positive) is the number of attacks correctly classified as attacks;

(2) FP (False Positive) is the number of normal records incorrectly classified as attacks;

(3) TN (True Negative) is the number of normal records correctly classified as normal records;

(4) FN (False Negative) is the number of attacks incorrectly classified as normal records.

4.4. Experimental Results and Discussion. In this section, our heterogeneous detection model is compared with Random Forest, *k*-NN, and Bagging comprising the component classifiers when the three algorithms are used by themselves. Here, our comparisons are based on SVD and un-SVD, respectively.

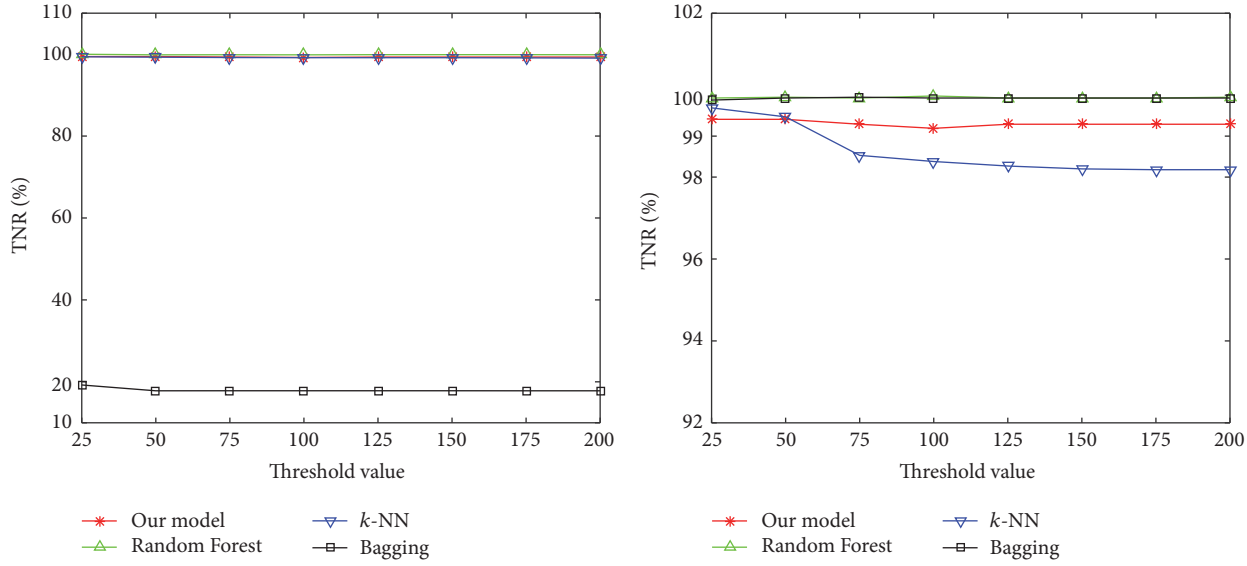
We refer to the past experience threshold value along with conducting many experiments. In this paper, we finally select eight threshold values to evaluate the performance of our model. Experimental results demonstrate that TNR, accuracy, and precision of our model are excellent in the detection model, and the model is more stable than the previous three algorithms in TNR, accuracy, and precision.

In Figure 2(a), when Random Forest, *k*-NN, and Bagging are processed by SVD, respectively, our detection model is compared with them for TNR. It is shown that the TNRs of our model are about 99.4% for different threshold values. However, the TNRs of Random Forest, *k*-NN, and Bagging are about 99.8%, 99.1%, and 17.8%, respectively. Therefore, the TNR of our model is very close to the TNR of Random Forest, and it is superior to the TNRs of *k*-NN and Bagging.

In Figure 2(b), when Random Forest, *k*-NN, and Bagging construct alone their component classifiers by un-SVD, our detection model is compared with them for TNR. It is shown that the TNRs of Random Forest and Bagging are about 99.9% and 99.9%, respectively. The TNR of *k*-NN falls from 99.6% to 98.1% when the range of threshold value is from 25 to 200. The experimental results demonstrate that the TNR of our model is close to the TNRs of Random Forest and Bagging, and it is superior to the TNR of *k*-NN. In addition, the TNR of *k*-NN is relatively unstable with the change of different threshold values.

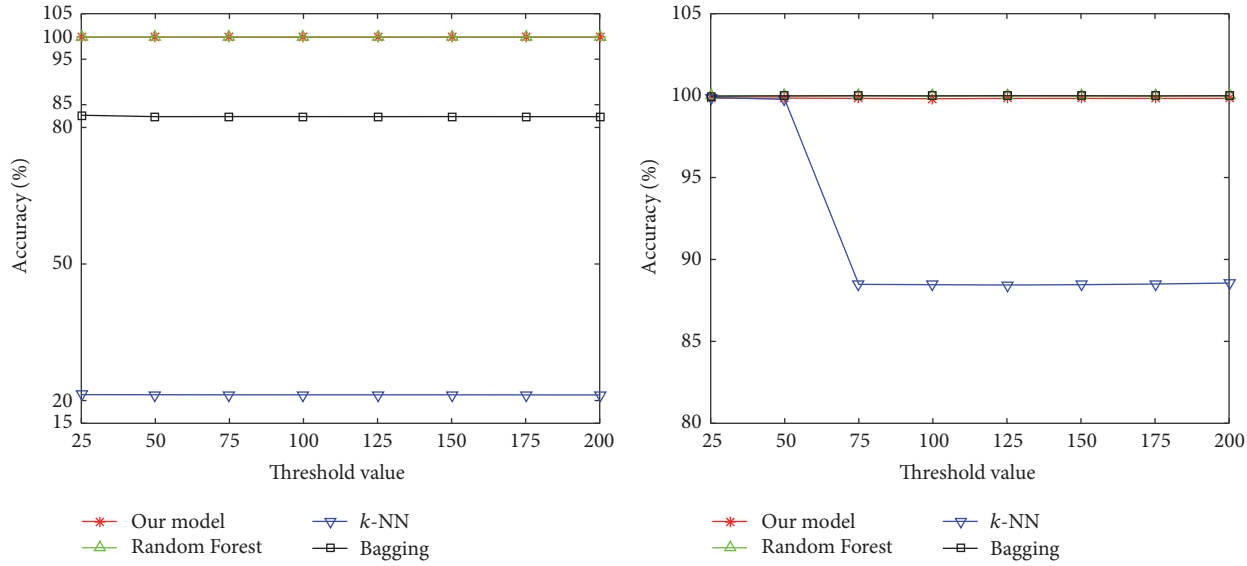
In Figure 3(a), when Random Forest, *k*-NN, and Bagging are processed by SVD, respectively, our detection model is compared with them for accuracy. It is shown that the accuracies of our model are about 99.8%. However, the accuracies of Random Forest, *k*-NN, and Bagging are about 99.9%, 21.2%, and 82.3%, respectively. Therefore, the accuracy of our model is very close to the accuracy of Random Forest, and it is superior to the accuracies of *k*-NN and Bagging.

In Figure 3(b), when Random Forest, *k*-NN, and Bagging construct alone their component classifiers by un-SVD, our detection model is compared with them for accuracy. It is shown that the accuracies of Random Forest and Bagging are about 99.9% and 99.9%, respectively. The accuracy of *k*-NN falls from 99.89% to 88.48% when the range of threshold value is from 25 to 75. The experimental results demonstrate that the accuracy of our model is close to the accuracies of Random Forest and Bagging, and it is superior to the accuracy of *k*-NN. In addition, the accuracy of *k*-NN is relatively unstable with the change of different threshold values.



(a) TNR comparison based on SVD (b) TNR comparison based on un-SVD

Figure 2: TNR for comparing our model with the other algorithms.

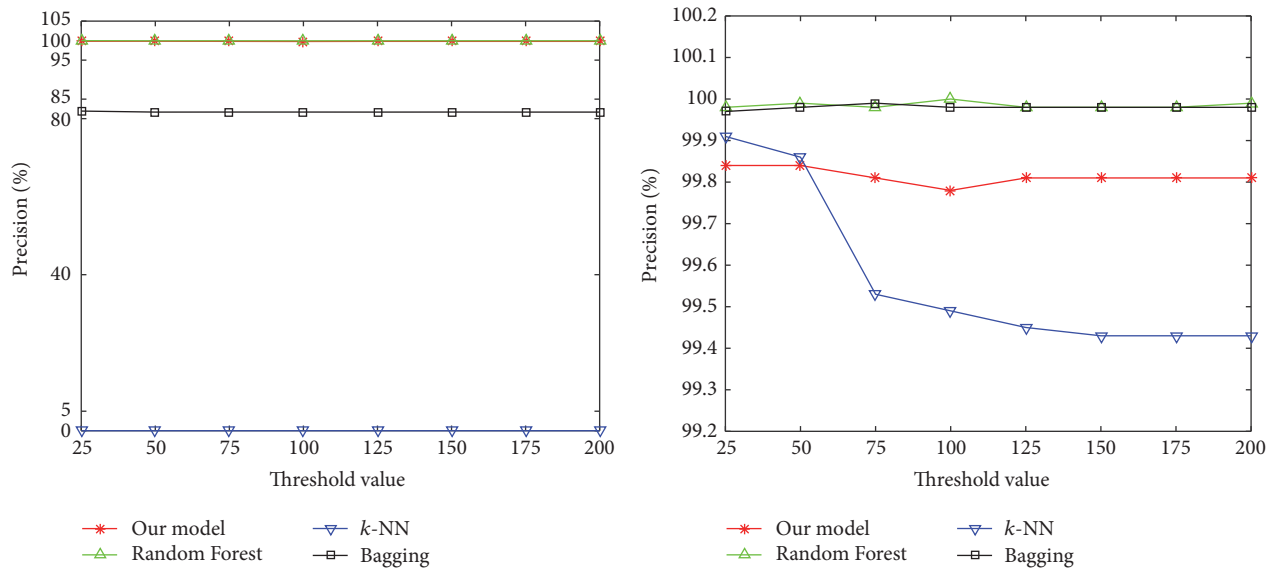


(a) Accuracy comparison based on SVD (b) Accuracy comparison based on un-SVD

Figure 3: Accuracy for comparing our model with the other algorithms.

In Figure 4(a), when Random Forest, k-NN, and Bagging are processed by SVD, respectively, our detection model is compared with them for precision. It is shown that the precisions of our model are about 99.84%. However, the precisions of Random Forest, k-NN, and Bagging are about 99.9%, 0, and 81.6%, respectively. Therefore, the precision of our model is very close to the precision of Random Forest, and it is superior to the precisions of k-NN and Bagging.

In Figure 4(b), when Random Forest, k-NN, and Bagging construct alone their component classifiers by un-SVD, our detection model is compared with them for precision. It is shown that the precisions of Random Forest and Bagging are about 99.98% and 99.98%, respectively. The precision of k-NN falls from 99.9% to 99.5% when the range of threshold value is from 25 to 75. The experimental results demonstrate that the precision of our model is close to the precisions of Random Forest and Bagging, and it is superior to the precision of k-NN. In addition, the precision of k-NN is relatively unstable with the change of different threshold values.



(a) Precision comparison based on SVD

(b) Precision comparison based on un-SVD

Figure 4: Precision for comparing our model with the other algorithms.

5. Conclusions

The efficient and exact DDoS attack detection is a key problem for diversity and complexity of attack traffic in high-speed Internet environment. In this paper, we study the problem from the perspective of hybrid heterogeneous multiclassifier ensemble learning. What is more, in order to get the stronger generalization and the more sufficient complementarity, we propose a heterogeneous detection system model, and we construct the component classifiers of the model based on Bagging, Random Forest, and k-NN algorithms. In addition, we design a detection algorithm based on SVD in heterogeneous classification ensemble model. Experimental results show that our detection method is excellent and is very stable in TNR, accuracy, and precision. Therefore, our algorithm and model have good detective performance for DDoS attack.

Competing Interests

The authors declare that they have no competing interests.

Acknowledgments

The work in this paper is supported by the Joint Funds of National Natural Science Foundation of China and Xinjiang (Project U1603261).

References

[1] F. Palmieri, S. Ricciardi, U. Fiore, M. Ficco, and A. Castiglione, "Energy-oriented denial of service attacks: an emerging menace for large cloud infrastructures," *The Journal of Supercomputing*, vol. 71, no. 5, pp. 1620–1641, 2015.

[2] Q. Yan, F. R. Yu, Q. X. Gong, and J. Q. Li, "Software-defined networking (SDN) and distributed denial of service (DDoS) attacks in cloud computing environments: a survey, some research issues, and challenges," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 1, pp. 602–622, 2016.

[3] J. Luo, X. Yang, J. Wang, J. Xu, J. Sun, and K. Long, "On a mathematical model for low-rate shrew DDoS," *IEEE Transactions on Information Forensics and Security*, vol. 9, no. 7, pp. 1069–1083, 2014.

[4] P. Xiao, W. Y. Qu, H. Qi, and Z. Y. Li, "Detecting DDoS attacks against data center with correlation analysis," *Computer Communications*, vol. 67, pp. 66–74, 2015.

[5] A. Saied, R. E. Overill, and T. Radzik, "Detection of known and unknown DDoS attacks using artificial neural networks," *Neurocomputing*, vol. 172, pp. 385–393, 2016.

[6] S. S. Mao, L. Xiong, L. C. Jiao, S. Zhang, and B. Chen, "Isomerous multiple classifier ensemble via transformation of the rotating forest," *Journal of Xidian University*, vol. 41, no. 5, pp. 48–53, 2014.

[7] L. I. Kuncheva and C. J. Whitaker, "Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy," *Machine Learning*, vol. 51, no. 2, pp. 181–207, 2003.

[8] J. J. Rodríguez, L. I. Kuncheva, and C. J. Alonso, "Rotation forest: a new classifier ensemble method," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 10, pp. 1619–1630, 2006.

[9] Z. L. Fu and X. H. Zhao, "Dynamic combination method of classifiers and ensemble learning algorithms based on classifiers Combination," *Journal of Sichuan University (Engineering Science Edition)*, vol. 43, no. 2, pp. 58–65, 2011.

[10] L. I. Kuncheva, "That elusive diversity in classifier ensembles," in *Proceedings of the 1st Iberian Conference on Pattern Recognition and Image Analysis (ibPRIA '03)*, pp. 1126–1138, Springer, Mallorca, Spain, June 2003.

[11] L. I. Kuncheva and J. J. Rodríguez, "Classifier ensembles with a random linear oracle," *IEEE Transactions on Knowledge and Data Engineering*, vol. 19, no. 4, pp. 500–508, 2007.

[12] J. F. Díez-Pastor, J. J. Rodríguez, C. García-Osorio, and L. I. Kuncheva, "Random balance: ensembles of variable priors classifiers for imbalanced data," *Knowledge-Based Systems*, vol. 85, pp. 96–111, 2015.

[13] L. Breiman, "Bagging predictors," *Machine Learning*, vol. 24, no. 2, pp. 123–140, 1996.

[14] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.

[15] D. T. Larose, "*k*-nearest neighbor algorithm," in *Discovering Knowledge in Data: An Introduction to Data Mining*, pp. 90–106, 2nd edition, 2005.

[16] W. Wang, X. Zhang, S. Gombault, and S. J. Knapskog, "Attribute normalization in network intrusion detection," in *Proceedings of the 10th International Symposium on Pervasive Systems, Algorithms, and Networks (I-SPAN '09)*, pp. 448–453, IEEE, Kaohsiung, Taiwan, December 2009.

[17] R. Lippmann, J. W. Haines, D. J. Fried, J. Korba, and K. Das, "The 1999 DARPA off-line intrusion detection evaluation," *Computer Networks*, vol. 34, no. 4, pp. 579–595, 2000.

[18] J. McHugh, "Testing Intrusion detection systems: a critique of the 1998 and 1999 DARPA intrusion detection system evaluations as performed by Lincoln Laboratory," *ACM Transactions on Information and System Security (TISSEC)*, vol. 3, no. 4, pp. 262–294, 2000.

[19] K. C. Khor, C. Y. Ting, and S. Phon-Amnuaisuk, "A cascaded classifier approach for improving detection rates on rare attack categories in network intrusion detection," *Applied Intelligence*, vol. 36, no. 2, pp. 320–329, 2012.

[20] P. Prasenna, R. K. Kumar, A. V. T. R. Ramana, and A. Devanbu, "Network programming and mining classifier for intrusion detection using probability classification," in *Proceedings of the International Conference on Pattern Recognition, Informatics and Medical Engineering (PRIME '12)*, pp. 204–209, IEEE, Tamilnadu, India, March 2012.

[21] C. Bae, W. C. Yeh, M. A. M. Shukran, Y. Y. Chung, and T. J. Hsieh, "A novel anomaly-network intrusion detection system using ABC algorithms," *International Journal of Innovative Computing, Information and Control*, vol. 8, no. 12, pp. 8231–8248, 2012.

[22] C. Guo, Y. Zhou, Y. Ping, Z. Zhang, G. Liu, and Y. Yang, "A distance sum-based hybrid method for intrusion detection," *Applied Intelligence*, vol. 40, no. 1, pp. 178–188, 2014.