Support Vector Machine Intrusion Detection Scheme Based on Cloud-Fog Collaboration

Ruizhong Du[1,2], Yun Li[1(✉)], Xiaoyan Liang[1,2], and Junfeng Tian[1,2]

[1] Cyberspace Security and Computer College, Hebei University, Baoding 071002, China
drzh@hbu.edu.cn, 15232045203@163.com

[2] Key Laboratory on High Trusted Information System in Hebei Province, Baoding 071002, China

Abstract. Fog computing is a new computing paradigm in the era of the Internet of Things. Aiming at the problem that fog nodes are closer to user equipment, with heterogeneous nodes, limited storage capacity resources, and greater vulnerability to intrusion, a lightweight support vector machine intrusion detection model based on Cloud-Fog Collaboration (CFC-SVM) is proposed. Due to the high dimensionality of network data, first, Principal Component Analysis (PCA) is used to reduce the dimensionality of the data, eliminate the correlation between attributes and reduce the training time. Then, in the cloud server, a support vector machine (SVM) optimized by the particle swarm algorithm is used to complete the training of the dataset, obtain the optimal SVM intrusion-detection classifier, send it to the fog node, and carry out attack detection at the fog node. Experiments with the classic KDD CUP 99 dataset show that the model in this paper is better than other similar algorithms in regard to detection time, detection rate and accuracy, which can effectively solve the problem of intrusion detection in the fog environment.

Keywords: Cloud-fog collaboration · Intrusion detection · Support vector machine · Particle swarm optimization

1 Introduction

The rapid development of the Internet of Things technology, the popularity of 5G networks, the emergence of a large number of industrial Internets, and augmented reality/virtual reality have increased network transmission capacity requirements, data distribution processing capabilities and real-time performance [1]. A single cloud-computing architecture cannot meet heterogeneous, low-latency, dense network access and services. Fog computing [2] allocates computing, communication, control and storage resources and services to users or devices and systems close to users to extend the cloud-computing mode to the edge of the network and provide network users with low latency, more flexible access and more secure network communication services. Fog and cloud computing do not have a mutual replacement relationship but have complementary infrastructure.

D. Wang et al. (Eds.): SPNCE 2020, LNICST 344, pp. 321–334, 2021.
https://doi.org/10.1007/978-3-030-66922-5_22

Together, the two form a computing model that is more suitable for the application scenario of the Internet of Things.

Compared to traditional cloud computing, the new features of fog computing [3] also create entirely new security and privacy protection issues. First, due to the heterogeneity of fog nodes, the whole fog computing system, including network facilities, service facilities, virtualization facilities and user terminals, is jointly owned by multiple owners. The result of this situation is that any part of the whole fog environment can be the target of network attack and privacy theft. Second, since the nodes providing computing and storage capabilities are deployed on the side of the network that is closer to the user, the nodes mainly provide services to nearby users [4]. This deployment and service method is a double-edged sword. On the one hand, it limits the scope of the network attack to a smaller range. On the other hand, once the attacker successfully controls the node, the whole region that the node serves will face risks. Common network threats include denial-of-service attacks [5], man-in-the-middle attacks, and probe attacks. Intrusion detection is the second security barrier behind the firewall, which can quickly detect security risks in the network. Traditional single-intrusion detection schemes have problems such as slow detection speed and low accuracy and are no longer adapted to the requirements of fog computing and edge computing environments [6] in response speed and high real-time performance. Aiming at this problem, this paper proposes a lightweight intrusion-detection model based on cloud-fog collaboration.

We will first analyze the security issues of the fog environment. Based on the resource constraints of fog nodes, a cloud-fog collaboration support vector machine intrusion detection model is proposed. This design makes full use of the advantages of the fog node and the cloud server and trains the classifier for detection in the cloud. After training, the training model is sent to the fog node. Our contributions in this study can be summarized as follows.

(1) Based on the classic three-layer architecture of fog computing, a lightweight intrusion detection model based on cloud-fog cooperative support vector machines is proposed.
(2) The principal component analysis process is added in the data processing stage to reduce the complexity of high-dimensional data by eliminating the correlation between features to reduce the dimension, making the data more lightweight and adapting to the fog nodes with limited resource storage capabilities.
(3) Particle swarm optimization is added to select SVM parameters to speed up training.
(4) By comparing the performance of the proposed scheme with the ELM, AdaBoost, and decision tree classifiers, the superiority of the proposed scheme is verified.

The rest of the paper is organized as follows. In Sect. 2, we will review related works on fog computing security and intrusion detection. In Sect. 3, we propose a support vector machine intrusion detection scheme based on cloud-fog collaboration. In Sect. 4, we introduce the SVM-based intrusion detection algorithm for fog computing. Section 5 will describe the simulation to verify the algorithm. The paper is concluded in Sect. 6.

2 Related Works

Currently, there are many studies analyzing the security threats and development directions of fog computing and mobile edge computing. Among them, Elazhary et al. [7] analyzed the intersections between the prominent areas of cloud computing and mobile computing research in detail to make their definitions more standardized. Parikh et al. [8] compared the structural advantages of cloud computing, edge computing, and fog computing with the possible security threats. Saad et al. [9] pointed out that the fog computing system can process a large amount of data locally, is completely portable, and can be installed on heterogeneous hardware. These characteristics make the fog computing platform very suitable for processing time- and location-sensitive applications. [10, 11] proposed fog computing as a new computing paradigm that can provide computing, storage and network services between users and traditional cloud platforms and described in detail the security issues in several different scenarios of fog computing, such as intelligent instrument certification and MIM attacks.

In recent years, there have been many studies on the security of fog computing systems. Among them, Razouk et al. [12] proposed a secure middleware architecture suitable for fog computing that provided computing power and secure communication links. This architecture focuses on ensuring the security of the communication process, and threats from outside the system are not considered. Hosseinpour et al. [13] proposed a distributed lightweight intrusion detection system based on an artificial immune system (AIS) suitable for fog computing, but the research did not perform in-depth analysis on the detection rate and accuracy of the system. Peng et al. [14] proposed a decision tree intrusion-detection system based on the characteristics of a large amount of data generated in a fog environment. The system did not consider the real-time requirements of fog computing and the limited storage capacity of fog nodes. Zhou et al. [15] applied the concept of fog computing to DDoS mitigation, distributing traffic monitoring and analysis work to local devices, and coordinating and merging work to the cloud center server to achieve rapid response in the case of a low false alarm rate.

Many scholars have proposed intrusion-detection schemes based on the characteristics of fog nodes. Among them, An et al. [16] fully considered the advantages of cloud servers and fog nodes and proposed an intrusion-detection scheme based on sample selection. This scheme shortened the training time but did not consider the problem of the limited storage capacity of fog nodes. S. Prabavathy et al. [17] proposed an intrusion-detection scheme based on an extreme learning machine (ELM) for the IoT environment of fog computing, which improved the accuracy and detection rate of intrusion detection but lacked consideration of real-time detection.

The support vector machine (SVM) is a small sample of machine learning methods based on statistical learning theory that has the characteristics of fast training speed and strong generalization ability and is suitable for intrusion detection in fog environments. From the current research results [18–21], the SVM algorithm has shown good results in various types of classification systems.

3 Support Vector Machine Intrusion Detection Scheme Based on Cloud-Fog Collaboration

The intrusion behavior of fog computing mainly comes from the terminal device layer. Therefore, the intrusion detection of fog computing is the process of discriminating and classifying the network data from the terminal device layer to determine whether it is an intrusion behavior. The fog node has fast calculation speed and limited resource storage capacity, and it is difficult to store a large number of training samples. To make reasonable use of the resources in the fog computing system and effectively perform intrusion detection, a collaborative intrusion-detection scheme for cloud servers and fog nodes is designed. The process is shown in Fig. 1:

1) Establish a communication link: Due to the heterogeneity of user terminals, the fog node provides network connections to each terminal device through different protocols and collects the data generated by each terminal device in real time.
2) The cloud server preprocesses and trains the original data: The entire training set is stored in the cloud server, and the entire training process is completed in the cloud server to generate a training model and save it.
3) The fog node sends a detection instruction: After the fog node establishes a communication link with the terminal device, it collects a large amount of network data generated by the terminal device and sends a detection instruction to the cloud server.
4) The cloud server sends the training model: After receiving the detection instruction, the cloud server sends the data processing model and the trained classification detection model to the fog node.
5) Fog node detection process: After receiving the model, the fog node uses the model for data processing and detection and generates detection results.
6) Intrusion response: The detected abnormal data are sent to the intrusion response module, and the intrusion response module performs the corresponding processing.

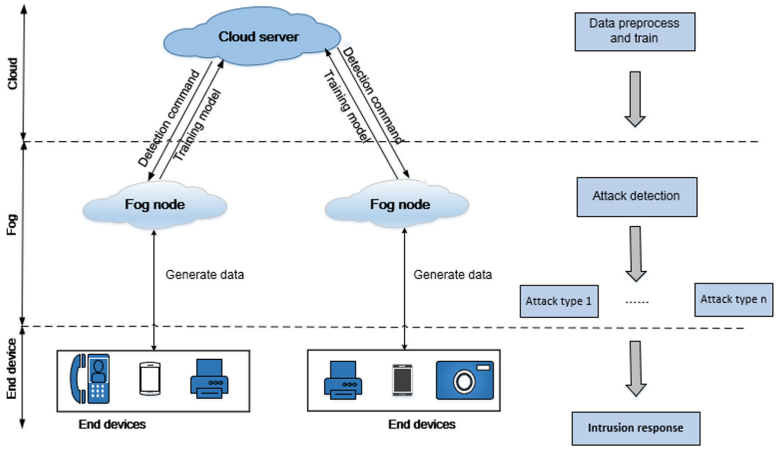

Fig. 1. Cloud-fog collaboration intrusion detection scheme

4 PCA and PSO-Optimized SVM for Intrusion Detection

4.1 PCA Data Dimensionality Reduction

Network data generally have high dimensional characteristics, resulting in a longer training time required for intrusion detection. Therefore, before training, dimension-reduction processing is performed on the training data to eliminate the correlation between attributes and maximize the retention of original data information. The core idea of principal component analysis (PCA) is to transform a set of variables that may be related to a set of linearly uncorrelated variables through orthogonal transformation. This paper uses PCA to make dimensional reduction on the training data.

This method can reduce the storage cost while removing the correlation of different dimensions. The contribution rate and cumulative contribution rate data of each eigenvalue are shown in Fig. 2. The value of the main ordinate axis represents the proportion of the variance value of each principal component in the total square difference after dimension reduction. The larger the proportion is, the more important the principal component is. The value of the secondary ordinate axis represents the cumulative value of the proportion of the variance value of each principal component to the total square difference, that is, the cumulative contribution rate.

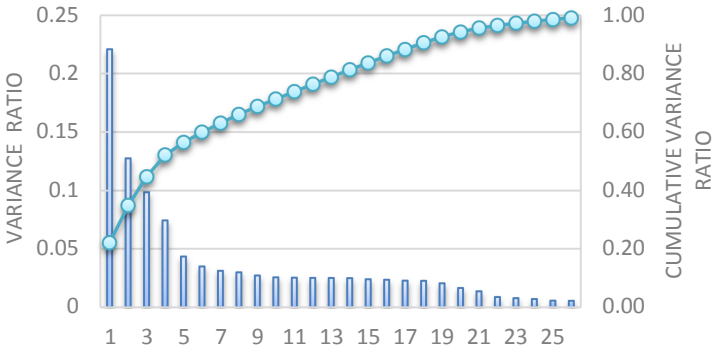

Fig. 2. Principal component contribution rate and cumulative contribution rate

4.2 SVM Algorithm

SVM (Support Vector Machine) was first proposed by Vapnik et al. The second kind is the problem that does not have linear separability. The SVM algorithm can use a kernel function to map the nonlinear separable dataset to high-dimensional space so that it has separability in high-dimensional space to achieve classification.

The calculation flow of the SVM algorithm is as follows: for linearly separable sample set $T = \{(x_1, y_1), (x_2, y_2), \cdots, (x_l, y_l)\}$, construct and solve the optimization problem of variables w and b:

$$\min J = \frac{\|w\|^2}{2} + C\sum_{i=1}^{l} \xi_i$$

$$\text{s.t. } y_i(wx_i + b) \geqslant 1 - \xi_i$$

$$\xi > 0_i, i = 1, 2, \cdots, l \tag{1}$$

where w is the weight vector, b is the threshold value, C is the penalty factor, which is used to punish the wrong samples, and ξ_i is the slack variable. Equation (7) is transformed into its dual problem by using Lagrange dual theory:

$$\max Q(\alpha) = \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i=1}^{l} \sum_{j=1}^{l} \alpha_i \alpha_j y_i y_j \langle x_i x_j \rangle$$

$$s.t. \sum_{i=1}^{l} \alpha_i y_i = 0, \quad \alpha_i > 0, i = 1, 2, \cdots, l \tag{2}$$

where α_i, α_j are the LaGrange multipliers. Using the quadratic programming method in operational research to solve Eq. (8), the optimal classification function is obtained as follows:

$$f(x) = \text{sgn}\left[\sum_{i=1}^{l} \alpha_i^* y_i \langle x^\star x_i \rangle + b^* \right] \tag{3}$$

where the sgn$(\cdot)$ function is used to identify the sample category, α_i^* is the solution of formula (8), and $\alpha_i^\star \neq 0$, $b^\star$ is the classification threshold. For the linear nonseparable type in this paper, the kernel function $K(x_i, x_j)$ satisfying the Mercer condition should be used to map the sample to the high-dimensional space to transform it into the linear separable problem. After mapping to the high-dimensional space, the corresponding dual problem becomes:

$$\max Q(\alpha) = \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i=1}^{l} \sum_{j=1}^{l} \alpha_i \alpha_j y_i y_j K(x_i, x_j)$$

$$\text{s. t.} \sum_{i=1}^{l} \alpha_i y_i = 0, \quad 0 \leqslant \alpha_i \leqslant C, i = 1, 2, \cdots, l \tag{4}$$

Finally, the optimal classification function is obtained as follows:

$$f(x) = \text{sgn}\left[\sum_{i=1}^{l} \alpha_i^* y_i K(x^\star, x_i) + b^* \right] \tag{5}$$

In this paper, the radial basis kernel function RBF is used, where g is the kernel parameter.

$$K(x_i, x_j) = \exp\left(-\frac{\|x_i - x_j\|^2}{2\sigma^2} \right) = \exp\left(-g \|x_i - x_j\|^2 \right) \text{ g} > 0 \tag{6}$$

4.3 PCA and PSO-Optimized Support Vector Machine for Intrusion Detection

The specific steps of PCA and PSO-optimized support vector machine for intrusion detection are as follows:

(1) Use principal component analysis to analyze the original data, and the first k principal components with a cumulative contribution rate greater than 99% are used as input variables, which are imported into the support vector machine model for simulation and prediction.
(2) Initialize the particle swarm position and velocity.
(3) Calculate the fitness value of all particle swarms. The fitness function is $F = \sum_{i=1}^{n} (y_i - \overline{y_i})^2$, where y_i and $\overline{y_i}$ are the actual output value and the expected output value of SVM model training, respectively.
(4) Evaluate population U_{id}, find the individual extreme value and global extreme value, and update the particle velocity and position.
(5) Determine whether the end condition of optimization is met. If it is, the optimization is ended, and the optimal solution is found. If not, return to step (3).
(6) Give the optimal parameters are given to the SVM model. The trained model is used to detect the fog nodes.

5 Experimental Evaluation

This simulation environment is performed under a Windows 10 64-bit operating system, i5 3.20 GHz processor, 16 GB memory environment, and the SVM algorithm is implemented in MATLAB (R2013a).

There is currently no dataset specifically for fog computing intrusion detection. This article uses the public intrusion detection dataset KDD CUP 99 23 to verify the effectiveness of the proposed scheme.

The dataset mainly includes four types of attack types: denial-of-service (DOS), remote-to-local (R2L), user-to-root (U2R), and port monitoring and scanning attacks (probing). The public dataset provides multiple datasets for training and testing. This paper uses the corrected dataset, which contains a total of 303,736 datasets. The distribution of various attack types in the corrected dataset is shown in Fig. 3. In experiment 1, this dataset was split into two mutually exclusive sets using the "hold-out" method, where the set containing 80% of the data was used as the training set S, and the other set containing 20% of the data was used as the test set T. T is used to evaluate the test error as an estimate of the generalization error.

5.1 Data Preprocessing

The corrected dataset includes 41 attributes, and the data format is as follows: 2, tcp, smtp, SF, 1684, 363, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0.00, 0.00, 0.00, 0.00, 1.00, 0.00, 0.00, 104, 66, 0.63, 0.03, 0.01, 0.00, 0.00, 0.00, 0.00, 0.00, and normal. Before application, data preprocessing is required. The process of data preprocessing is as follows:

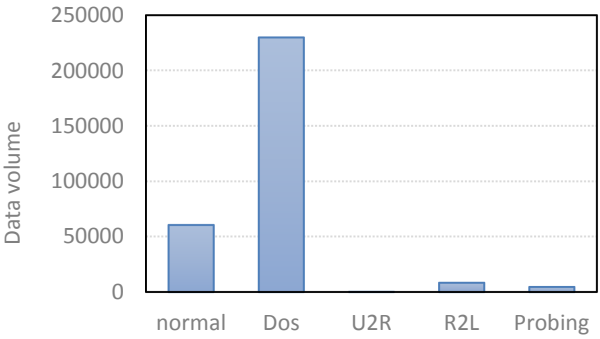

Fig. 3. Various types of data distribution maps corrected

Numeralization
Transform the symbol attributes protocol_type, service, flag in the training set and test set into numeric types. The category labels are converted into numerical representations. The specific assignment of the digitized category labels is shown in Table 1.

Table 1. 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
Probe = 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

Standardization
Standardize the data obtained in (1) to eliminate the influence of different dimensions on the calculation results.

$$new\_data = \frac{ori\_data - ori\_avg}{ori\_std} \tag{7}$$

The *new_data* in the above formula represents the standardized data, *ori_data* represents the original data, *ori_avg* represents the vector of the mean of each dimension on the original dataset, and *ori_std* represents the vector of the variance of each dimension.

Normalization
Normalize each standardized data to [0, 1] with the min-max method, as follows:

$$v' = \frac{v - min_i}{max_i - min_i} \tag{8}$$

where v is a value of the ith attribute column, min_i is the minimum value of the ith attribute column, and max_i is the maximum value of the ith attribute column.

5.2 Experimental Evaluation Index

As shown in Table 2, TP (True Positive) indicates the number of abnormal samples correctly classified as abnormal samples, TN (True Negative) indicates the number of normal samples classified as normal samples, FP (False Positive) indicates the number of normal samples that are incorrectly classified as abnormal samples, and FN (False Negative) indicates the number of incorrect classifications of abnormal samples into normal samples.

Table 2. Confusion matrix of classification results

Confusion matrix		Predict	
		Attack (positive)	Normal (negative)
True value	Attack (positive)	TP	FN
	Normal (negative)	FP	TN

To verify the effectiveness of the scheme, this paper uses calculation time, precision, detection rate (DR), accuracy (Acc), and false alarm rate (FAR) as the measurement algorithm performance indicators as follows:

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

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

$$\text{Acc} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{11}$$

$$\text{FAR} = \frac{\text{FP}}{\text{TN} + \text{FP}} \tag{12}$$

5.3 Experimental Analysis

To verify the validity of the CFC-SVM intrusion detection model, two experiments are designed in this paper.

Experiment 1: compare the effect of SVM and other classifiers after PCA dimensionality reduction;

Experiment 2: verify the effect of cloud server and fog node collaborative detection on detection time.

The SVM algorithm uses radial basis kernel functions. Penalty parameter C and kernel function parameter g are obtained by the iterative optimization of particle swarm

optimization. The parameters of the particle swarm algorithm are set as follows: the number of iterations is 100, and the number of populations is 50. The final parameter optimization results are C = 19.528 and g = 1.238.

The first experiment compares a single SVM algorithm, an extreme learning machine (ELM) algorithm that retains 99% of the information through PCA dimensionality reduction, an Adaboost algorithm, and a decision tree algorithm. To prove the superiority of the PCA-SVM classifier, the experiment was conducted 10 times, and 20% of the data in the dataset was randomly selected as the test set each time. The average of the experimental results was 10 times. Table 3 shows the average value of the data of various attack types in the second experimental test set. Table 4 shows the comparison of PCA-SVM and SVM detection effects. It can be seen from the figure that the dimensionality reduction operation does not reduce the SVM detection effect while retaining the original dataset information and greatly accelerates the training time.

Table 3. Average value of various attack types in 10 experimental test sets

Records	Test
Normal	12,050
DOS	44,772
Probe	465
R2L	1,165
U2R	8
Total	58,460

Table 4. Comparison of SVM and PCA-SVM

	Running time (s)	Normal		DOS		R2L		U2R		Probe	
		DR (%)	FAR (%)	DR (%)	FAR (%)	DR (%)	FAR (%)	DR (%)	FAR (%)	DR (%)	FAR (%)
SVM	9.12	99.8	0.12	99.5	1.57	80.2	2.59	21.2	5.89	91.2	3.56
PCA-SVM	6.53	99.7	0.12	99.5	1.55	81.3	2.58	20.6	5.88	92.3	3.12

Figure 4 shows the comparison of the detection precision between the SVM classifier and other classifiers when PCA dimension reduction is used to retain 99% of the original dataset information. As seen from Fig. 4, after PCA dimension reduction, SVM classification has better detection accuracy for various attack types than Adaboost, ELM, and decision tree.

Figure 5 shows the comparison of the detection rate between the SVM classifier and other classifiers when PCA is also used to reduce the dimensionality and retain 99% of the original dataset information. The PCA-SVM classifier has a high detection rate for

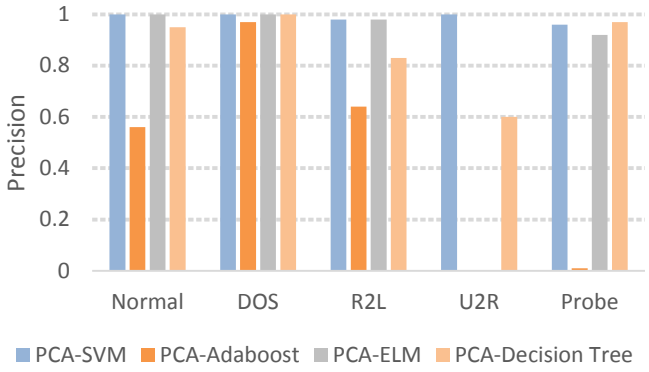

Fig. 4. Precision contrast of four kinds of attacks in four modes.

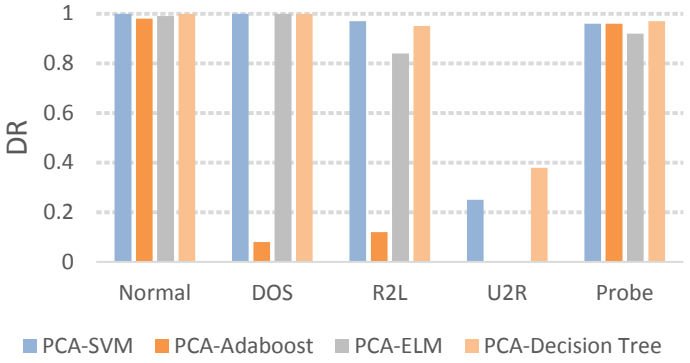

Fig. 5. Detection rate contrast of four kinds of attacks in four modes.

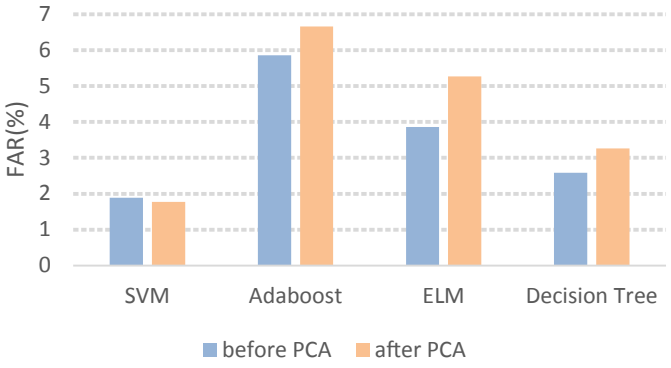

Fig. 6. Comparison of false alarm rate of four classifiers before and after PCA

the R2L, DOS, probe and Normal types. Because the content of U2R attack type data in the dataset is very low, the detection rate of U2R is low.

Figure 6 shows the comparison of the false alarm rate of four classifiers before and after dimension reduction. It can be seen from the figure that the false alarm rate of the SVM classifier decreases after dimension reduction by PCA, while that of the other three classifiers increases after dimension reduction. After PCA dimensionality reduction, SVM is better than the other algorithms in terms of precision, detection rate and false alarm rate.

Experiment 2 is used to verify the impact of the intrusion-detection scheme proposed in this paper, which is collaborative between the cloud server and fog node, on the detection time. MATLAB is used for simulation. SVM is deployed in the cloud server, and 10000 pieces of data are randomly selected as the detection samples. Ten experiments are conducted, and the comparison parameters are the average value of detection time and detection accuracy.

Table 5. Comparison of different algorithms in accuracy and detection time

Algorithm	Accuracy (%)	Detection time (s)
CFC-SVM	98.5	4.1
SVM	96.5	4.5
ELM	96.1	8.8
Decision tree	84.5	12.7

According to the results in Table 5, the PCA-SVM algorithm has relatively high accuracy and less detection time when applied to intrusion detection in a fog environment. The main reason is that the dimensionality reduction of data makes the calculation speed faster. The particle swarm optimization algorithm is applied to the selection of SVM parameters, which increases the training speed. In the fog environment, data training in the cloud server and detection in the fog end are adopted. The layered model can greatly reduce the detection time.

6 Conclusions

The network where the fog nodes are located is highly dynamic and highly real time, which presents a new challenge to the intrusion detection of fog computing. This paper proposes a cloud-fog cooperative intrusion detection model, which is added during pre-processing PCA dimensionality reduction, eliminating the correlation between various dimensions of network data and reducing the storage and calculation overhead. The dataset is trained in the cloud server with an SVM classifier (where the selection of SVM parameters is optimized by particle swarm optimization) and saving the training model greatly saves the detection time at the fog nodes. The proposed model is verified on the KDD CUP 99 dataset, and several experiments have demonstrated the advantages of this model from different perspectives and have contributed to solving the problems caused by resource constraints in fog computing.

Acknowledgment. This project is supported by Natural Science Foundation of China (No. 61572170, No.61170254), The Key Projects of Natural Science Foundation of Hebei Provience (No. F2019201290), The Natural Science Foundation of Hebei Province (No. F2018201153) and Hebei Natural Science Foundation under grant No. F2018201197, Research on Impreciseness Analysis Key Technology of Network Access Control Intention. We hereby express our thanks.

References

1. Shi, W., Cao, J., Zhang, Q., Li, Y., Xu, L.: Edge computing: vision and challenges. IEEE Internet Things J. **3**(5), 637–646 (2016)
2. Liu, C., Xiang, F., Wang, P., Sun, Z.: A review of issues and challenges in fog computing environment. In: DASC/PiCom/DataCom/CyberSciTech, pp. 232–237 (2019)
3. Puliafito, C., Mingozzi, E., Longo, F., Puliafito, A., Rana, O.: Fog computing for the internet of things: a survey. ACM Trans. Internet Tech. **19**(2), 18:1–18:41 (2019)
4. Oma, R., Nakamura, S., Duolikun, D., Enokido, T., Takizawa, M.: An energy-efficient model for fog computing in the Internet of Things (IoT). Internet of Things **1–2**, 14–26 (2018)
5. Puthal, D., Mohanty, S.P., Bhavake, S.A., Morgan, G., Ranjan, R.: Fog computing security challenges and future directions [energy and security]. IEEE Consum. Electron. Mag. **8**(3), 92–96 (2019)
6. Noura, H.N., Salman, O., Chehab, A., Couturier, R.: Preserving data security in distributed fog computing. Ad Hoc Netw. **94**, 101937 (2019)
7. Elazhary, H.: Internet of Things (IoT), mobile cloud, cloudlet, mobile IoT, IoT cloud, fog, mobile edge, and edge emerging computing paradigms: disambiguation and research directions. J. Netw. Comput. Appl. **128**, 105–140 (2019)
8. Parikh, S., Dave, D., Patel, R., Doshi, N.: Security and privacy issues in cloud, fog and edge computing. In: EUSPN/ICTH 2019, pp. 734–739 (2019)
9. Khan, S., Parkinson, S., Qin, Y.: Fog computing security: a review of current applications and security solutions. J. Cloud Comput. **6**, 19 (2017). https://doi.org/10.1186/s13677-017-0090-3
10. D'Souza, C., Ahn, G.-J., Taguinod, M.: Policy-driven security management for fog computing: Preliminary framework and a case study. In: IRI 2014, pp. 16–23 (2014)
11. Ficco, M.: Internet-of-Things and fog-computing as enablers of new security and privacy threats. Internet Things **8** (2019)
12. Razouk, W., Sgandurra, D., Sakurai, K.: A new security middleware architecture based on fog computing and cloud to support IoT constrained devices. In: IML 2017, pp. 35:1–35:8 (2017)
13. Hosseinpour, F., Amoli, P.V., Plosila, J., et al. An intrusion detection system for fog computing and iot based logistic systems using a smart data approach. Int. J. Digit. Content Technol. Appl. **10**(5) (2016)
14. Peng, K., Leung, V.C.M., Zheng, L., Wang, S., Huang, C., Lin, T.: Intrusion detection system based on decision tree over big data in fog environment. Wirel. Commun. Mob. Comput. **2018** (2018)
15. Zhou, L., Guo, H., Deng, G.: A fog computing based approach to DDoS mitigation in IIoT systems. Comput. Secur. **85**, 51–62 (2019)
16. An, X., Zhou, X., Lü, X., Lin, F., Yang, L.: Sample selected extreme learning machine based intrusion detection in fog computing and MEC. Wirel. Commun. Mob. Comput. **2018** (2018)
17. Prabavathy, S., Sundarakantham, K., Shalinie, S.M.: Design of cognitive fog computing for intrusion detection in internet of things. J. Commun. Netw. **20**(3), 291–298 (2018)

18. Liu, Y., Bi, J.-W., Fan, Z.-P.: A method for multi-class sentiment classification based on an improved one-vs-one (OVO) strategy and the support vector machine (SVM) algorithm. Inf. Sci. **394**, 38–52 (2017)
19. Cui, J., Shi, G., Gong, C.: A fast classification method of faults in power electronic circuits based on support vector machines. Nephron Clin. Pract. **24**(4), 701–720 (2017)
20. Li, L.: Analysis and data mining of intellectual property using GRNN and SVM. Pers. Ubiquit. Comput. **24**(1), 139–150 (2019). https://doi.org/10.1007/s00779-019-01344-8
21. Gu, J., Wang, L., Wang, H., Wang, S.: A novel approach to intrusion detection using SVM ensemble with feature augmentation. Comput. Secur. **86**, 53–62 (2019)
22. Subba, B., Biswas, S., Karmakar, S.: Enhancing performance of anomaly based intrusion detection systems through dimensionality reduction using principal component analysis. In: ANTS 2016, pp. 1–6 (2016)
23. KDD CUP 99 data set. http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html