Check for updates

Security and Privacy in Internet of Things - Research Article

International Journal of
Distributed
Sensor Networks

International Journal of Distributed Sensor Networks
2020, Vol. 16(11)
© The Author(s) 2020
DOI: 10.1177/1550147720971517
journals.sagepub.com/home/dsn
SAGE

An efficient method for network security situation assessment

Xiaoling Tao[1,2], Kaichuan Kong[1], Feng Zhao[1], Siyan Cheng[3] and Sufang Wang[1]

Abstract

Network security situational assessment, the core task of network security situational awareness, can obtain security situation by comprehensively analyzing various factors that affect network status. Thus, network security situational assessment can provide accurate security state evaluation and security trend prediction for users. Although plenty of network security situational assessment methods have been proposed, there are still many problems to solve. First, because of high dimensionality of input data, computational complexity in model construction could be very high. Moreover, most of the existing schemes trade computational overhead for accuracy. Second, due to the lack of centralized standard, the weights of indicators are usually determined empirically or by subjective opinions of domain expert. To solve the above problems, we propose a novel network security situation assessment method based on stack autoencoding network and back propagation neural network. In stack autoencoding network and back propagation neural network, to reduce the data storage overhead and improve computational efficiency, we use stack autoencoding network to reduce the dimensions of the indicator data. And the low-dimensional data output by hidden layer of stack autoencoding network will be the input data of the error back propagation neural network. Then, the back propagation neural network algorithm is adopted to perform network security situation assessment. Finally, extensive experiments are conducted to verify the effectiveness of the proposed method.

Keywords

Network security, situation assessment, dimension reduction, stack autoencoding network, back propagation neural network

Date received: 11 August 2020; accepted: 16 October 2020

Handling Editor: Yanjiao Chen

Introduction

With the prevalence of big data, the amount of services provided by Internet witnesses an explosive growth.[1] This is due to the extension of Internet application and the integration of various fields, such as national defense, military, and public transportation. However, network security incidents occur frequently and the techniques used in network attacks become more and more complex. As a result, how to accurately and effectively evaluate security status has become a hot research topic in the field of network security, which is related to the stability and security of network

[1] Guangxi Cooperative Innovation Center of Cloud Computing and Big Data, Guilin University of Electronic Technology, Guilin, China
[2] Guangxi Key Laboratory of Cryptography and Information Security, Guilin, China
[3] Viterbi School of Engineering, University of Southern California, Los Angeles, CA, USA

Corresponding author:
Feng Zhao, Guangxi Cooperative Innovation Center of Cloud Computing and Big Data, Guilin University of Electronic Technology, Guilin 541004, China.
Email: fengzhao@guet.edu.cn

operation.[2] Therefore, it is necessary to adopt holistic approach to effectively deal with situational awareness data. Thus, network security situational awareness (NSSA) emerges.[3] NSSA[4] was first proposed by T bass, provides decision-makers with knowledge of the most critical assets, threats, and related vulnerabilities, and effective countermeasures and risk mitigation technologies to correctly and timely response to threats.[5]

Network security situation assessment is the core of NSSA technology, which can comprehensively analyze all kinds of uncontrollable security factors and provide information about current network security situation. When network threats come in, proactive defense measures are taken to ensure timely protection of network security. Network security situation assessment determines the performance of techniques for NSSA, and it is of great importance to comprehensively understand the state of network environment, the ability to detect network security, and handle network threat events.

Network security situation assessment is a crucial part of network security, and it is an useful technique to understand the status and performance of network, which is important to the management of networks. Network security situation assessment has been applied to many fields, such as electric power information network,[6] naval systems,[7] aviation cyber security,[8] and vehicle network.[9]

Existing network security situation assessment methods can be summarized into the following categories, that is, methods based on mathematical model (MM), approaches based on knowledge reasoning (KR), and methods based on pattern recognition (PR).[10] As a PR-based method, back propagation neural network (BPNN) has a certain flexible network structure and a strong non-linear mapping ability. According to the specific situation, the number of intermediate layers and the number of neurons in each layer can be arbitrarily set. However, existing NSSA data exhibit characteristics, such as complex structure, multi-source, and massive. As a result, high dimensionality of input data will lead to high complexity of model construction, huge CPU costs in model training, slow training speed, and numerous parameters, which will ultimately affect the efficiency of the method. Therefore, it is necessary to perform data dimensionality reduction to avoid curse of dimensionality, improve computational efficiency, and reduce probability of overfitting.

The main contributions of this article are summarized as follows:

1. Stack autoencoding network (SAE) is used to reduce the dimensionality of non-linear data and the complexity of model construction before performing security situation assessment;
2. Loss function is used to determine the number of self-coding network layers to ensure information integrity of the data;
3. BPNN is adopted to perform network security situation assessment, and contextual relevance of network security is fully taken into consideration.

The remainder of this article is organized as follows. In section "Related work," we review the related work, and give some preliminaries in section "Preliminaries." Then, we propose the network security situation assessment method based on SAE + BPNN in section "Proposed model." In section "Experimental study," we briefly introduce the experimental environment and data we used, and the experimental results are analyzed in detail. Finally, we conclude the article in section "Conclusion."

Related work

The network security situation assessment methods are usually divided into MM, KR, and PR. The evaluation method based on MM considers various factors to evaluate the situation, which aims to evaluate the network situation from different angles.

Chen et al.[11] established a hierarchical network security threat situation quantitative assessment model based on the bottom-up, local first and global strategy. Their proposed method calculates the threat indicator by weighting the importance of attacks, services, hosts, and the whole network layer by layer, thus evaluating the security threat situation. Li et al.[12] used fuzzy c-mean clustering and optimal clustering criteria to process the data, thus obtaining the optimal clustering center and number of clusters. Moreover, they combined analytic hierarchy process (AHP) to establish a multi-factor two-level assessment model to obtain the final situation assessment result. Wang et al.[6] proposed a hierarchical chaos simulation annealing (CSA) method based on AHP and gray cluster analysis (GCA). In their method, they used AHP to build a hierarchical CSA to determine the weight of every threat. Meanwhile, GCA is used to build the standard layer of the indexing system. Bian et al.[13] proposed a multi-level fuzzy comprehensive network security situation evaluation model based on improved AHP and fuzzy comprehensive evaluation method. Note that traditional network situation assessment methods cannot effectively assess the security situation of distributed denial of service (DDoS) attacks. Zhang et al.[14] proposed a DDoS attack security situation assessment model based on the fusion features of fuzzy clustering algorithms. Their model can reasonably and effectively evaluate the security status of DDoS attacks. Meanwhile, their model is more flexible than non-fuzzy methods. However, the methods based on MM rely on expert knowledge in the process of index selection, index weight determination, and model construction. As a result, the evaluation results are easily affected by subjective factors.

The KR-based network security situation assessment methods assume that there is certain degree of correlation between the network security situation and the state of the network, which is susceptible to the influence of historical and current information. It uses theory of evidence, mathematical statistics, and fuzzy theory to learn historical prior information and current information to infer the current security status of the network. Based on semi-supervised naive Bayes (NB) classifiers, Xu et al.[15] proposed an improved algorithm based on the confidence of data classification, which can achieve situational classification of air combat data. Jin et al.[16] proposed a network security situation assessment model based on random forest (RF). Their method is based on the idea of multiple classifier combination, which consists of decision trees. Each tree depends on an independent sample, and all the trees have the same random vector distribution value in a forest. To effectively evaluate the impact of DDoS attacks on the network situation, Li et al.[17] computed the indicators representing the network situation in each layer, and then the indicators are fused with Dempster–Shafer (D-S) evidence theory to evaluate the impact. Fu et al.[18] improved the optimal fuzzy gray model using modified gray model (GM) (1,1) with residuals, and the optimal fuzzy gray model is used to the prediction of network security situation assessment.

The Markov model has important applications in network security situation assessment. Schemes[19,20] fully consider the interaction between the attacker and the defender, and proposed a network security assessment model based on the Markov decision process and game theory. To solve the problem that hidden Markov model (HMM) parameters are difficult to configure, Li and Li[21] proposed an improved situational assessment method based on HMM, which establishes HMM by obtaining the observation sequence and combines the improved simulated annealing (SA) algorithm with the Baum–Welch (BW) algorithm. HMM parameters are optimized, and the security situation value of the network is obtained by the method of quantitative analysis, which more accurately reflects the security situation of the network. Liu and Liu[22] used attack graphs to describe the causality of attack behaviors and combined the HMM to establish the probability mapping between the observation sequence and the attack state. Moreover, the Viterbi algorithm was used to calculate the maximum probability state transition sequence. Li and Zhoa[23] pointed out that the network evaluation time period is greatly affected by human, and the HMM state transfer matrix and observation symbol matrix are often determined empirically. To solve the above problems, they used sliding time window mechanism to extract observation values, and the hybrid multi-population genetic algorithm is adopted to train the HMM model parameters to improve accuracy. Although the method based on KR performs well when analyzing security problems on small dataset with low dimensionality, its evaluation efficiency is relatively low when dealing with massive high-dimensional data.

The PR-based network security situation assessment methods assume that the network security situation results can be obtained according to the degree of matching data. PR-based methods divide different security situation levels by learning the characteristics of data and use the data to match each of the divided results, thus obtaining the network security situation. To obtain a global optimal solution, Shi and Chen[24] proposed a twin support vector machines (SVM) model for command information system security situation sample data learning and parameter estimation, so as to evaluate the command information system security situation. Gao et al.[25] proposed an artificial fish swarm algorithm to optimize the information system security risk assessment model of SVM. The proposed method used the artificial fish swarm algorithm to optimize the penalty coefficient C and kernel function of SVM. The experimental results show that the method has high accuracy and convergence speed. Song et al.[26] proposed an information security situation assessment model based on genetic algorithm to optimize weights and thresholds of BPNN. Compared with the standard BPNN, BPNN optimized by genetic algorithm (GA-BP) neural network has lower simulation error and better fitting effect. Li et al.[27] and Dong et al.[28] use cuckoo search algorithm to optimize BPNN neural network parameters to avoid BP falling into local extreme values, thereby improving training speed and evaluation accuracy. Compared with genetic-BPNN algorithm, it has training time, error, and accuracy. Even better, Luo and Liu[29] used the rough intensive reduction attribute to take the membership of the samples calculated by fuzzy method as the input of neural network and the expert value as expected output of the network to improve training speed and accuracy. Zhang et al.[30] proposed a situation assessment method based on deep autoencoding network for the dependence of BPNN methods on label data. The deep autoencoder (AE) is used as the basic unit to construct a deep AE network, and the deep AE network is trained with expert experience and hierarchical evaluation methods to form a model with the ability to accurately evaluate the input situation data. However, the classification results are mostly obtained through machine learning techniques, and the middle part of the algorithm is difficult to follow.

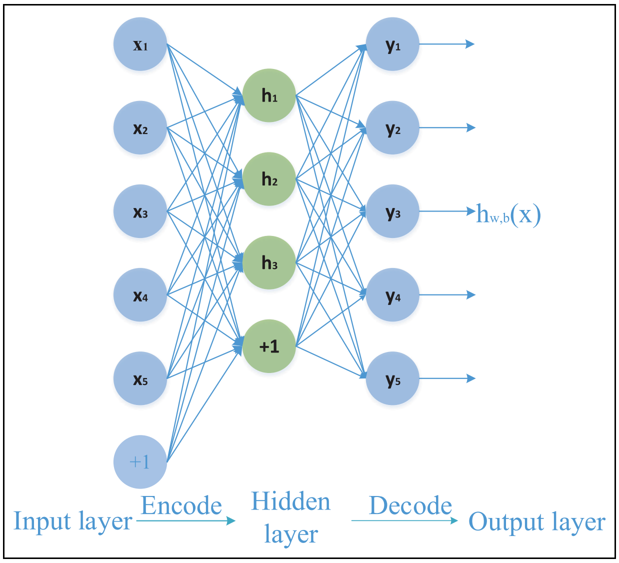

Figure 1. Self-encoding neural network.

Preliminaries

SAE

AE is an unsupervised learning algorithm driven by input data, which performs feature extraction on data through self-supervision without labels, thus resulting in data with reduced dimensionality, that is, less but more important features. The self-encoding neural network maps the input data to the hidden layer to realize data encoding. Then, the corresponding decoded data are obtained by mapping the encoded data, and the decoded data are regarded as the output data.

The encoder is composed of input layer, hidden layer, and output layer, in which the learning process from input layer to hidden layer is called encoding process, while from hidden layer to output layer is called decoding process, as shown in Figure 1.

From a general perspective, the input layer is $x = [x_1, x_2, \ldots, x_n]^T \in R^{n \times 1}$, the hidden layer is $h = [h_1, h_2, \ldots, h_d]^T \in R^{d \times 1}$, and the output layer is $y = x = [x_1, x_2, \ldots, x_n]^T \in R^{n \times 1}$. We define the weight matrix from the input layer to the hidden layer as $W(W \in R^{n \times d})$, the bias as $b = [b_1, b_2, \ldots, b_d]^T \in R^{d \times 1}$, the weight matrix from the hidden layer to the output layer as $W'(W' \in R^{n \times d})$, and the bias as $b' = [b'_1, b'_2, \ldots, b'_n]^T \in R^{d \times 1}$.

The output of self-encoding hidden layer can be expressed as

$$h = f_1(Wx + b) \tag{1}$$

where f_1 represents the activation function of the hidden layer. ReLu, Sigmoid, Tanh, etc. can be selected according to the specific application. The output of the self-encoding output layer can be expressed as

$$\hat{y} = f_2(W'x + b') \tag{2}$$

In the process of training the neural network, to reduce the parameters that need to be trained in the model, the following constraints are usually given

$$W = W' \tag{3}$$

At this point, it means that the learning model should contain three sets of parameters: W, b, b', and the parameter $\theta = \{W, b, b'\}$ adjustment of the learning model is mainly realized by minimizing the error function

$$\underset{\theta = \{W,b,b'\}}{\arg\min} \; [cost(x, \hat{y})] = 0 \tag{4}$$

1. When f_2 selects Sigmoid as the activation function, the error function of the self-encoding network can be expressed as

$$cost = -\sum_{i=1}^{n} [x_i log \hat{y}_i + (1 - x_i) log(1 - \hat{y}_i)] \tag{5}$$

2. When f_2 selects a linear function as the activation function, the error function of the self-encoding network can be expressed as

$$cost = ||x - \hat{y}||^2 \tag{6}$$

The overall error function can be expressed as

$$J_{AE}(\theta) = \sum_{x \in S} cost(x, f_2(f_1(x))) \tag{7}$$

The BP training is carried out using the stochastic gradient descent method combined with the error function to update the network parameters. The rules for parameter update are defined as follows (where η represents the learning rate)

$$W = W - \eta \frac{\partial cost(x, \hat{y})}{\partial W} \tag{8}$$

$$b = b - \eta \frac{\partial cost(x, \hat{y})}{\partial b} \tag{9}$$

$$b' = b' - \eta \frac{\partial cost(x, \hat{y})}{\partial b'} \tag{10}$$

SAE[31] is an improved research based on self-coding network, which is a kind of network constructed by connecting several ordinary self-encoder successively. As shown in Figure 2, it consists of several layers of self-encoding networks and a Softmax layer.

The training of stack self-encoding includes the following steps:

1. Input the original data, use an AE to train the input data to get the corresponding network

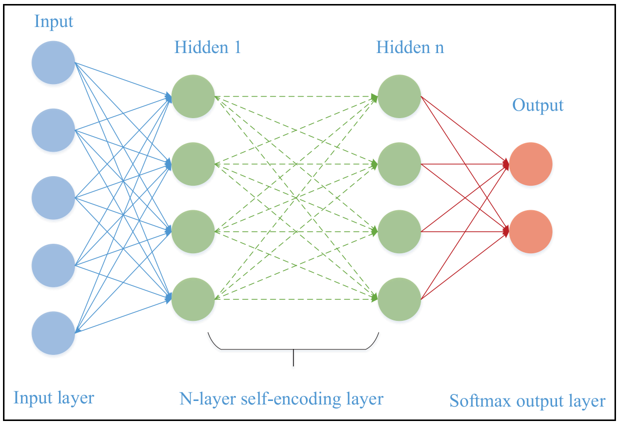

Figure 2. Deep self-coding neural network.

parameters, encode the original data through the trained AE network, and take the output result after encoding as the output result of the first hidden layer;

2. Take the output in Step 1 as the input and continue to use the training method in Step 1 to optimize and update the network parameters of this layer. Repeat this step until the last hidden layer is trained;
3. Take the output in Step 2 as the input and use the label corresponding of the original input data to train and optimize the network parameters of Softmax layer;
4. Calculate the loss cost function of all hidden layers and Softmax layers, and the partial derivative function value of each parameter in the network;
5. The initial network parameters calculated in Steps 1, 2, and 3 are taken as the initialization parameters of the whole network. Meanwhile, the loss cost functions and partial derivatives of the parameters obtained in Step 4 are used to calculate the updated network parameters and realize the parameter optimization of the whole network.

BPNN

The Error BPNN is also called BPNN, as shown in Figure 3. As a supervised learning algorithm, BPNN mainly uses the error function generated by the actual output value and the expected output value for BP, which adjusts connection weight and threshold parameters of neurons in each layer of the network. The training of network will be stopped and relevant parameters of the network will be saved by iterating the network with input data until the error function is reduced to the allowable range of the network.

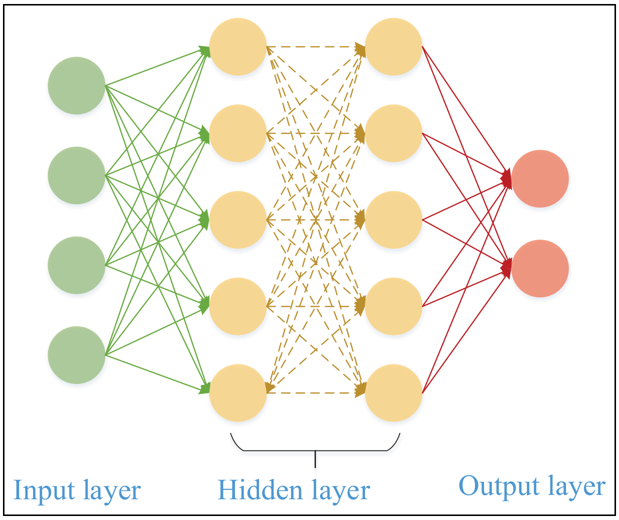

Figure 3. BPNN.

The specific training steps of BPNN are as follows:

1. Initialize the network. Assume that the input vector is $x = [x_1, x_2, \ldots, x_i, \ldots, x_m]^T \in R^{m\times l}$, the output vector is $y = [y_1, y_2, \ldots, y_i, \ldots, y_m]^T \in R^{m\times 1}$, and the output of each neuron in the lth hidden layer is: $h^{(l)} = [h_1^{(l)}, h_2^{(l)}, \ldots, h_j^{(l)}, \ldots, h_{S_l}^{(l)}]^T \in R^{S_l\times 1}$, where S_l is the number of neurons in the lth layer. Let $W_{ij}^{(l)}$ be the connection weight between the jth neuron of the $l-1$ layer and the ith neuron of the lth layer.
2. Calculate the output of each neuron in the hidden layer and the output layer. $b_i^{(l)}$ is the bias of the ith neuron in the lth layer, then

$$h_i^{(l)} = f(net_i^{(l)}) \tag{11}$$

$$net_i^{(l)} = \sum_{j=1}^{S_{l-1}} W_{ij}^{(l)} h_j^{(l-1)} + b_i^{(l)} \tag{12}$$

where $net_i^{(l)}$ is the input of the ith neuron in the lth layer, and $f(\cdot)$ is the activation function of the neuron. The non-linearity of neural network is mainly reflected in the selection of its activation function. When the linear activation function is adopted, the multi-layer neural network is equivalent to the complex linear function formed by the combination of multiple linear functions. In the process of selecting the activation function, a non-linear function can be taken to make the neural network have certain non-linear capability.

3. Calculate the error function of the output layer and the hidden layer. Given the training sample, let $m = \{(x(1), y(1)), (x(2), y(2)), \ldots, (x(m), y(m))\}$ and $d(i)$ be the expected output

generated by the input $x(i)$. BP algorithm adopts the gradient descent method to adjust the weight parameters of each hidden layer neuron to ensure that the actual output of the neural network is close to the expected output.

With batch update method, for the given training sample m, the error function is defined as

$$E = \frac{1}{m}\sum_{i=1}^{m} E(i) \tag{13}$$

where $E(i)$ is the training error of a single sample

$$E(i) = \frac{1}{2}\sum_{k=1}^{n} (d_k(i) - y_k(i))^2 \tag{14}$$

Sample population error

$$E(i) = \frac{1}{2m}\sum_{i=1}^{m}\sum_{k=1}^{n} (d_k(i) - y_k(i))^2 \tag{15}$$

4. Calculate new connection weights and thresholds

$$W_{ij}^{(l)} = W_{ij}^{(l)} - \alpha \frac{\partial E}{\partial W_{ij}^{(l)}} \tag{16}$$

$$b_i^{(l)} = b_i^{(l)} - \alpha \frac{\partial E}{\partial b_i^{(l)}} \tag{17}$$

where α is the learning rate and its value range is [0,1].

5. Repeat multiple iterations. Repeat Steps 2–4, adjust the network parameters according to the errors generated by each iteration, and finish the training and save the model parameters until all samples are trained or the error function has reached the preset range.

Proposed model

Network structure

Due to high dimensionality and complexity of data, existing evaluation methods based on neural network use multi-layer and multi-neuron networks. However, these methods are not efficient. In this article, we propose a network security situation evaluation method based on SAE and BPNN.

SAE is an unsupervised learning algorithm, which is mostly used in data denoising, sparse high-dimensional data dimensionality reduction, and so on. In network security situation assessment field, the indicator data are high dimensional and sparse. We use SAE to reduce data dimensionality while ensuring that there is no information loss in indicator data and combine BPNN to conduct network security situation assessment. Meanwhile, we select the commonly used security situation assessment methods, such as SVM and NB, for auxiliary verification. The experimental results show that the method has fast convergence rate in training phase and high accuracy in evaluation phase, which is convenient for administrators to understand the network security status accurately.

SAE-BPNN uses the coded data output from the last hidden layer of SAE network as the input of BPNN, which not only can ensure the non-linear relationship of data but also can reduce the dimension of input data, as shown in Figure 4.

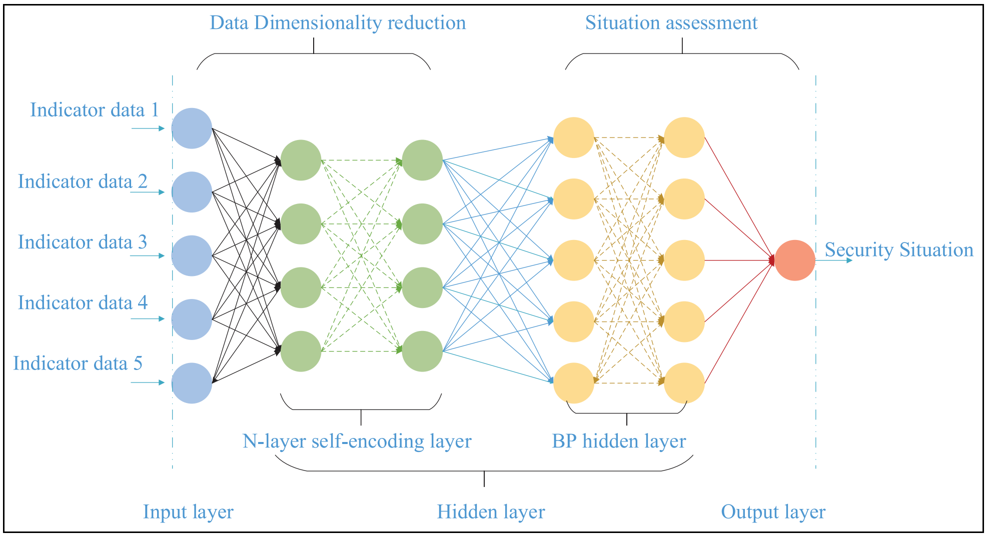

Figure 4. SAE-BPNN.

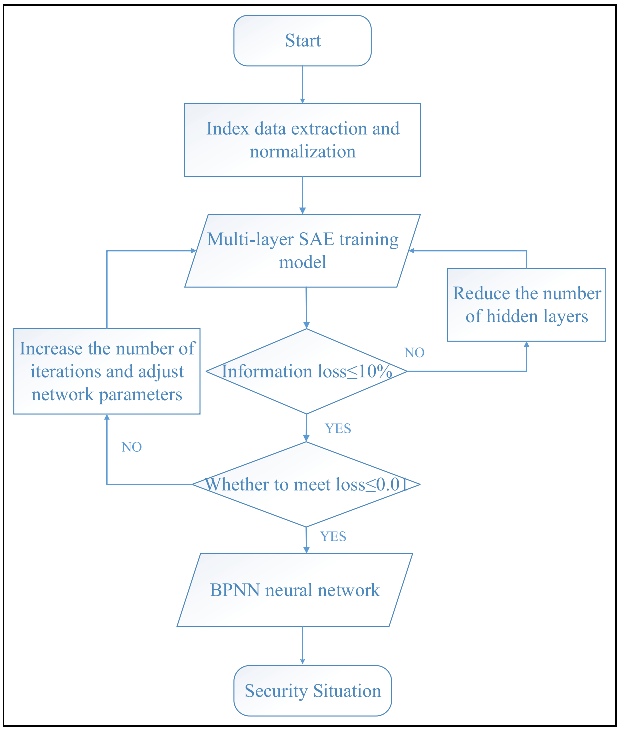
Start
Index data extraction and normalization
Multi-layer SAE training model
Reduce the number of hidden layers
Increase the number of iterations and adjust network parameters
Information loss≤10%
NO
YES
NO
Whether to meet loss≤0.01
YES
BPNN neural network
Security Situation

Figure 5. SAE-BPNN algorithm evaluation process.

The SAE-BPNN algorithm

The specific process of the SAE-BPNN evaluation method is as follows (see Figure 5).

1. *Indicator data extraction and normalization processing*: most of the NSSA data are generated in the form of network traffic, alarm logs, and so on. It is necessary to execute perception data extraction according to the indicator system and then normalize the indicator data according to the corresponding normalization criteria.
2. *SAE training model*: the SAE training model inputs the normalized indicator data. Specifically, it first determines the number of hidden layer in the model. Training through multi-layer SAE neural network, calculating the corresponding loss value, and determining whether within the scope of the reasonable losses. If not reasonable, adjusting the number of hidden layers. Then it needs to adjust the model parameters. It adjusts SAE model parameters and the number of iterations to obtain the model under the optimal error value and uses the final SAE model to encode the input data as the input of BPNN.

First, the combination formula of hidden layer and output layer is deduced according to the self-coding formula in section "SAE"

$$\begin{cases} h_1 = f_1(W_1x + b_1) \\ h_2 = f_1(W_2h_1 + b_2) \\ \vdots \\ h_n = f_1(W_nh_{n-1} + b_n) \end{cases} \tag{18}$$

$$\begin{cases} \hat{y}_1 = f_2\left(W'_1h_1 + b'_1\right) \\ \hat{y}_2 = f_2\left(W'_2h_2 + b'_2\right) \\ \vdots \\ \hat{y}_n = f_2\left(W'_nh_n + b'_n\right) \end{cases} \tag{19}$$

The number of layers of SAE is determined by the information loss rate.

The formula of single layer information loss rate is as follows

$$\text{Loss}_j = \sum_{j=1}^{n}\left|1 - \frac{\hat{y}_{ij}}{x_{ij}}\right| \times 100\% \tag{20}$$

The loss rate of n-layer comprehensive information is as follows

$$\text{Loss}_{\text{all}} = \prod_{i=1}^{n}\sum_{j=1}^{n}\left|1 - \frac{\hat{y}_{ij}}{x_{ij}}\right| \times 100\% = \sum_{j=1}^{n}\text{Loss}_j \tag{21}$$

where x_{ij} represents the jth input value of layer i network and y_{ij} represents the jth output value of layer i network. The number of network layers can be determined according to the loss value range.

Finally, the number of SAE layers and the output results of N layers are determined by the loss range

$$\begin{cases} h_n = f_1(W_nh_{n-1} + b_n) \\ \hat{y}_n = f_2\left(W'_nh_n + b'_n\right) \end{cases} \tag{22}$$

h_n will be used as the input of BPNN for the next step of calculation.

3. *BPNN situation assessment*: input the data after non-linear dimensionality reduction processing and its corresponding label into BPNN, thus obtaining the optimal model through multiple iterations and evaluating the security situation.

$h^{(0)} = h_n$ will be used as the input of BPNN for the next step of calculation

$$\begin{cases} net^{(1)} = W^{(l)}h^{(0)} + b^{(l)} \\ net^{(2)} = W^{(l)}h^{(1)} + b^{(2)} \\ \vdots \\ net^{(l)} = W^{(l)}h^{(l-1)} + b^{(l)} \end{cases} \tag{23}$$

$$y_{result} = h^{(l)} = f_3(net^{(l)}) \tag{24}$$

y_{result} is the final output of the model.

Table 1. CIDDS-001 data property table.

Serial No.	Name	Description
1	Src IP	Source IP address
2	Src Port	Source port
3	Dest IP	Destination IP address
4	Dest Port	Target port
5	Proto	Transport protocol (e.g. ICMP, TCP, or UDP)
6	Date first seen	First appearance of data
7	Duration	Duration of traffic
8	Bytes	Number of bytes transferred
9	Packets	Number of packets transmitted by the packet
10	Flags	The series of TCP flags included in the traffic
11	Class	Traffic category tag (normal, attacker, victim, suspicious, or unknown)
12	AttackType	Attack type (portScan, dos, bruteForce)
13	AttackID	Attack ID (All traffic data belonging to the same attack carry the same attack ID)
14	AttackDescription	Attack parameter information (e.g. the number of attempts to guess passwords for SSH brute force attacks)

Experimental study

Experimental environment

We conduct experiments on a machine equipped with NVIDIA TITAN XP GPU, with Ubuntu 18.04 operating system, Python 3.6, and PyCharm Community 2017.3. Meanwhile, we use TensorFlow 1.4.1, Keras library, and machine learning library scikit-learn for model training.

Experiment dataset

To verify the validity of the SAE-BPNN algorithm, we select the Coburg Intrusion Detection Dataset-001 (CIDDS-001)[32] of Coburg University of Technology as the research object. CIDDS is an evaluation dataset created based on an anomaly network intrusion detection system. The basic idea behind CIDDS is to use OpenStack to create tagged stream-based datasets in a virtual environment.

The network topology of CIDDS-001 dataset is divided into internal network and external network, as shown in Figure 6. The internal environment includes multiple clients and typical servers, such as e-mail servers and Web servers. Network attacks contain denial of service (DoS), brute force attacks, and port scans. Since the origin, target and timestamp of the attack being executed are known; it is easy to mark the recorded NetFlow data.

CIDDS-001 dataset has a total of 14 attributes, as shown in Table 1.

In this experiment, CIDDS-001 dataset's Week 2 external stream data are selected for analysis, and the external stream data flow attack on the second day is shown in Table 2.

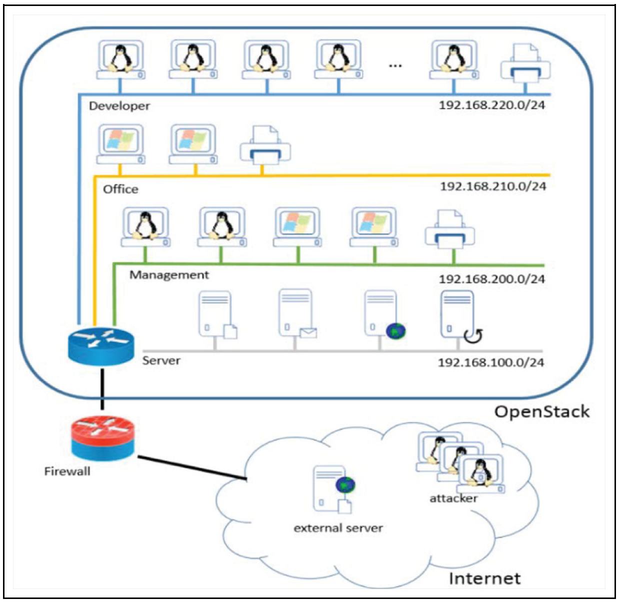

Figure 6. CIDDS-001 network topology.

The relevant information can be extracted from Table 2. Three attacks are initiated before 12 o'clock and after 12 o'clock. Therefore, the data stream after 12 o'clock is selected for training, and the data stream before 12 o'clock is used for testing; the ratio of training set to validation set is 2:1.

Classification of the normalization scheme of this experimental indicator system:

1. The maximum value of the six types of indicators (e.g. data stream duration, number of used protocols, number of source addresses, number of destination addresses, number of network ports, and type of data stream) are within a certain range. The normalization scheme uses the extreme value method

Table 2. Attack log information.

Attacker	Attack generation time	End time	Attack methods
ATTACKER1	2017/3/23 9:46	2017/3/23 9:48	portScan
ATTACKER1	2017/3/23 10:14	2017/3/23 10:30	portScan
ATTACKER1	2017/3/23 11:33	2017/3/23 12:00	bruteForce
ATTACKER1	2017/3/23 12:43	2017/3/23 12:48	bruteForce
ATTACKER1	2017/3/23 13:36	2017/3/23 13:42	bruteForce
ATTACKER1	2017/3/23 18:03	2017/3/23 18:07	bruteForce

Table 3. SAE network parameter configuration.

Activation function	Optimization function	Loss function	No. of iterations	Iteration step
Sigmoid	Adam	mean_ squared_ error	1000	8

$$\tilde{x}_i = \frac{x_i - x_{min}}{x_{max} - x_{min}} \tag{25}$$

where x_i is the current value of the indicator, and $\tilde{x}_i$ is the value after normalization of the indicator.

2. The number of transmitted packets, the number of transferred bytes, and the amount of suspicious data have a large amount of variation. As a result, their maximum value cannot be determined. Therefore, the inverse cotangent function method is adopted in the normalization scheme

$$\tilde{x}_i = \frac{arctg(x_i)}{\pi} + 0.5, (x_i \in R, \tilde{x}_i \in [-1, 1]) \tag{26}$$

or

$$\tilde{x}_i = \frac{arctg(x) * 2}{\pi}, (x_i \in R^+, \tilde{x}_i \in [-1, 1]) \tag{27}$$

where x_i is the current value of the indicator, and $\tilde{x}_i$ is the value after normalization of the indicator.

Experiment results

Dimension reduction part. We first need to determine the number of hidden layers in SAE. Then according to the SAE-BPNN evaluation process, we normalize the indicator data and use SAE to perform data dimensionality reduction on the training model, and the experimental parameter setting is given in Table 3.

First, we select the number of hidden layers for SAE. Considering the memory space occupied by the data storage after dimensionality reduction, the analysis is performed according to the theoretical space proportion, actual test proportion, and data proportion of the actual data store. The theoretical storage space of data is shown in Table 4, and the actual storage space of data in file storage is shown in Table 5. Note that the data are a float-64 type, and each one of the data occupies 8 bytes. The initial dimensionality is 9. Data footprint = (dimension × number of data pieces) × unit data occupies storage space. For example, 1000 data samples of 9-dimensional data take up $1000 \times 9 \times 8$ byte = 72,000 bytes. To compare the occupancy of data storage, the self-coded SAE of Layers 1, 2, and 3 is used to encode and reduce the dimension of indicator data, respectively. The input dimension is 9 and the output dimension is set to 4. Specifically, SAE input at Layer 1 to hidden layer is 9-4, the second layer is 9-7-4, and the third layer is 9-7-6-4.

Table 4. The theoretical space occupied by data storage.

	Dimensionality			
No. of data samples	9 (bytes)	7 (bytes)	6 (bytes)	4 (bytes)
1000	72,000	56,000	48,000	32,000
10,000	720,000	560,000	480,000	320,000
100,000	7,200,000	5,600,000	4,800,000	3,200,000

Table 5. The actual space occupied by data storage.

	Dimensionality			
No. of data samples	9 (bytes)	7 (bytes)	6 (bytes)	4 (bytes)
1000	73,728	69,632	69,632	65,536
10,000	712,704	651,264	618,496	602,112
100,000	7,127,040	6,545,408	6,189,056	6,078,464

As shown in Figure 7, we can find that during the actual data storage process, the data are stored in the excel file. When the data are reduced from 9 to 4 dimensions, the average storage space of 1000 pieces of data is reduced from 73,728 bytes to 65,536 bytes, saving nearly 15% of storage space.

Figure 8 shows the Loss caused by constructing Layers 1, 2, and 3 SAE hidden layer. According to the

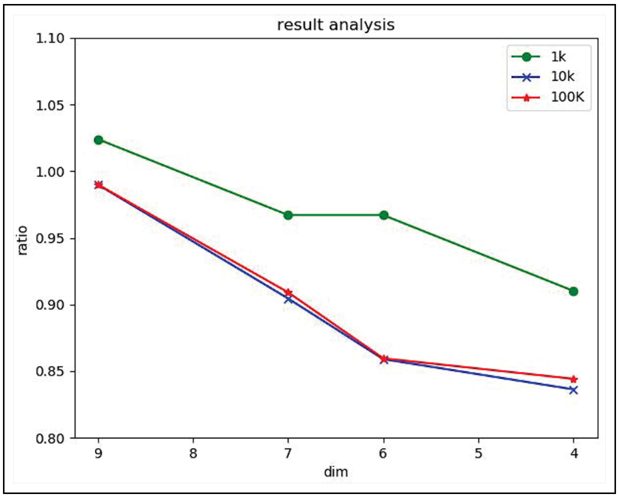

Figure 7. Ratio of actual storage space to ideal storage space.

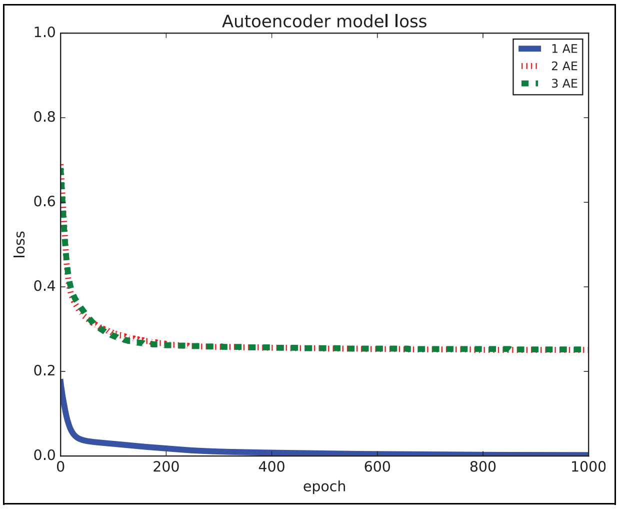

Figure 8. Loss comparison of SAE algorithm.

analysis results, it can be seen that when the indicator data are coded for dimensionless reduction, the Loss value is close to 0 when the number of SAE hiding layers is 1 and the number of iterations epoch is 400, which indicates that after the dimensionless reduction of SAE at Layer 1, the output data can better restore the input data, and the information integrity rate of the input data is close to 100%. When the number of hidden layers of SAE is 2 or 3, the Loss value tended to be stable when epoch = 200 times, but the Loss value remained above 0.2. Meanwhile, the SAE hidden layer output data are equivalent to the feature information that loses more than 20% of the original input data. Through experimental analysis, Layer 1 SAE is finally selected to encode the indicator data, and SAE iteration times are selected as 600.

Table 6. BPNN network parameter configuration.

Activation function	Optimization function	No. of iterations	Iteration step
Relu	Lbfgs	200	8

Evaluation. The data after SAE dimensionality reduction are input into BPNN, and the BPNN parameter settings are shown in Table 6. To verify the effectiveness of SAE + BPNN, BPNN and SAE are used for comparison to evaluate the network security situation. The test experiment selects the external data stream from 9 am to 12 am on the Tuesday of second week of CIDDS-001 dataset. There are three attacks between 9:46 and 9:48, 10:14 to 10:30, and 11:33 to 12:00, and the experimental comparison results are listed in Figure 9.

From Figure 9 we can see that although SAE can roughly determine the attack situation during the process of network security situation assessment, the evaluation results fluctuate relatively much. Moreover, BPNN can accurately detect the situation of attack, but there is a false positive within 120–140 min. SAE + BPNN can accurately determine the time of attack and its evaluation accuracy is the most accurate, which can exactly identify the attack time.

Evaluation performance analysis. Except the comparison experiment conducted by combining SAE with BP, we also select SVM and NP to analyze the security situation and verify the effectiveness of the proposed method.

From Table 7 we can easily find that the proposed method has a certain improvement in terms of accuracy as compared with BPNN, and the combination of SAE, NB, and SVM also improves the evaluation accuracy. Meanwhile, from Table 8 we can see that the running time of the methods after applying SAE dimensionality reduction is less than that of the methods BP, NB, and SVM without dimensionality reduction.

Conclusion

In this article, we propose a network domain security situation assessment method based on SAE-BPNN. First, the proposed method extracts the indicator data of network domain and normalizes them. Then, SAE is used for dimension reduction and feature extraction. Moreover, the network security situation value will be calculated by BPNN algorithm, which can evaluate the network domain security situation quantitatively. Finally, through a series of comparative experiments, we proved that the proposed method based on SAE

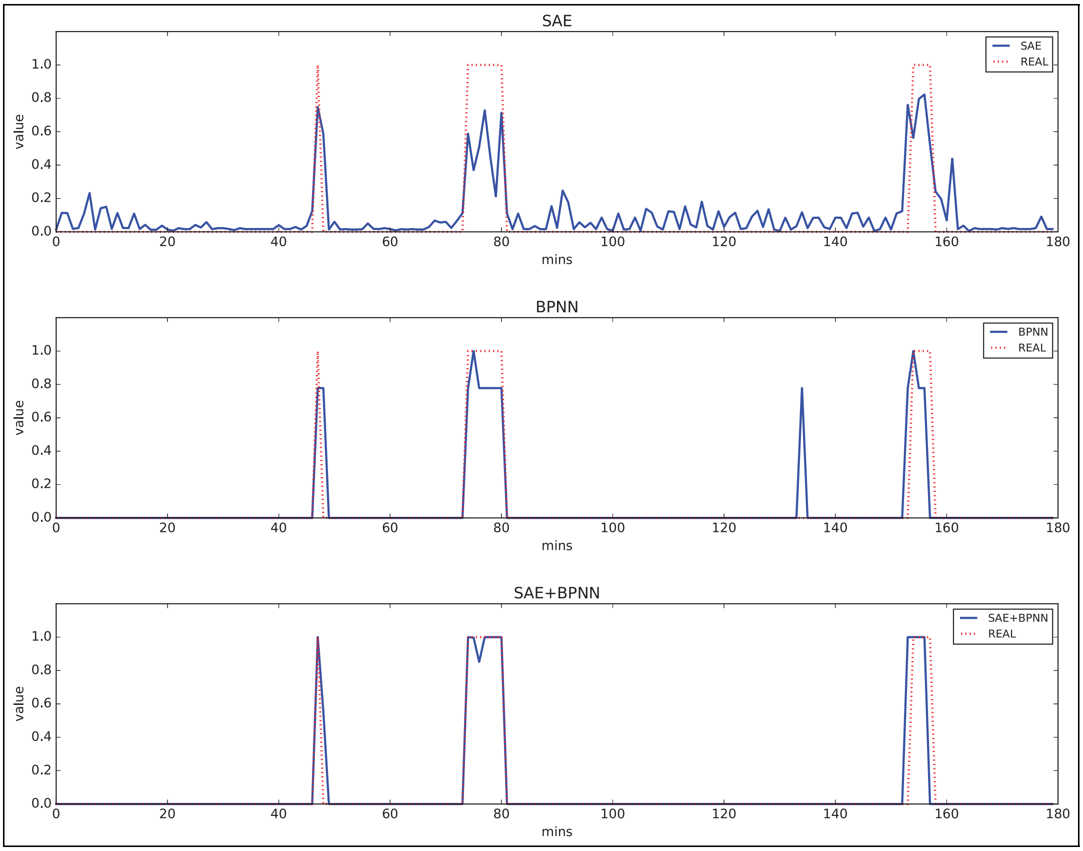

Figure 9. Comparison of SAE-BPNN experiments.

Table 7. Comparison of algorithm performance.

Evaluation algorithm	Evaluation index (%)		
	Precision rate (P)	Recovery rate (R)	Comprehensive evaluation index (F1)
BPNN	98.01	97.78	97.86
SAE + BPNN	99.05	98.89	98.93
NB	97.83	97.78	97.57
SAE + NB	98.33	97.78	97.92
SVM	96.78	96.67	96.14
SAE + SVM	97.13	97.22	97.17

BPNN: back propagation neural network; SAE: Stack autoencoding network; SVM: support vector machines.

Table 8. Comparison of algorithm time.

SAE dimension reduction	Evaluation algorithm (s)		
	BPNN	NB	SVM
No dimension reduction	0.029796	0.000884	0.001189
After SAE dimension reduction	0.026230	0.000726	0.001072

SAE: stack autoencoding network; BPNN: back propagation neural network; NB: naive Bayes; SVM: support vector machine.

and BPNN can accurately evaluate the security situation of network domain. And this method has the ability to reduce the dimensions of input data while preserving useful features of the data, which can reduce the storage overhead and computing resources and improve the evaluation efficiency.

Declaration of conflicting interests

The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Funding

The author(s) disclosed receipt of the following financial support for the research, authorship, and/or publication of this article: This work was supported by the National Natural Science Foundation of China (No. 61962015), the Natural Science Foundation of Guangxi (No. 2016GXNSFAA380098), and the Science and Technology Program of Guangxi (No. AB17195045).

ORCID iD

Xiaoling Tao iD https://orcid.org/0000-0002-6573-2291

References

1. He Z, Cai Z and Yu J. Latent-data privacy preserving with customized data utility for social network data. *IEEE T Veh Technol* 2018; 67(1): 665–673.
2. Lei H, Li HL, Wei ZH, et al. Summary of research on IT network and industrial control network security assessment. In: *2019 IEEE 3rd information technology, networking, electronic and automation control conference (ITNEC)*, Chengdu, China, 15–17 March 2019, pp.1203–1210. New York: IEEE.
3. Tianfield H. Cyber security situational awareness. In: *2016 IEEE international conference on internet of things (iThings) and IEEE green computing and communications (GreenCom) and IEEE cyber, physical and social computing (CPSCom) and IEEE smart data (SmartData)*, Chengdu, China, 15–18 December 2016, pp.782–787. New York: IEEE.
4. Bass T. Intrusion detection systems and multisensor data fusion. *Commun ACM* 2000; 43(4): 99–105.
5. Evesti A, Kanstrn T and Frantti T. Cybersecurity situational awareness taxonomy. In: *2017 international conference on cyber situational awareness, data analytics and assessment (Cyber SA)*, London, 19–20 June 2017, pp.1–8. New York: IEEE.
6. Wang YF, Wang J, Xu ZB, et al. Assessing cyber-threats situation for electric power information networks. In: *2013 ninth international conference on natural computation (ICNC)*, Shenyang, China, 23–25 July 2013, pp.1557–1562. New York: IEEE.
7. Jacq O, Brosset D, Kermarrec Y, et al. Cyber attacks real time detection: towards a cyber situational awareness for naval systems. In: *2019 international conference on cyber situational awareness, data analytics and assessment (Cyber SA), Oxford*, 3–4 June 2019, pp.1–2. New York: IEEE.
8. Kiesling T, Krempel M, Niederl J, et al. A model-based approach for aviation cyber security risk assessment. In: *2016 11th international conference on availability, reliability and security (ARES)*, Salzburg, 31 August–2 September 2016, pp.517–525. New York: IEEE.
9. Li R, Li F and Zhang J. Vehicle network security situation assessment method based on attack tree. *E&ES* 2020; 428(1): 012021.
10. Gong ZH and Zhuo Y. Research on network situation awareness. *J Softw* 2010; 21(7): 1605–1619.
11. Chen XZ, Zheng QH, Guan XH, et al. A quantitative evaluation method for the threat situation of hierarchical network security. *J Softw* 2006(4): 885–897.
12. Li FW, Yang SC and Zhu J. Improved network security situation assessment method based on fuzzy hierarchy method. *J Comput Appl* 2014; 34(9): 26222644–26222626.
13. Bian N, Wang X and Mao L. Network security situational assessment model based on improved AHP_FCE. In: *2013 sixth international conference on advanced computational intelligence (ICACI)*, Hangzhou, China, 19–21 October 2013, pp.200–205. New York: IEEE.
14. Zhang R, Cheng J and Tang X. DDoS attack security situation assessment model using fusion feature based on fuzzy C-means clustering algorithm. In: *2018 international conference on cloud computing and security*, Haikou, China, 8–10 June 2018, pp.654–669. New York: Springer.
15. Xu XM, Yang RN and Fu Y. Situation assessment for air combat based on novel semi-supervised naive Bayes. *J Syst Eng Electr* 2018; 29: 768–779.
16. Jin YH, Shen YJ, Zhang GD, et al. The model of network security situation assessment based on forest. In: *2016 7th IEEE international conference on software engineering and service science (ICSESS)*, Beijing, China, 26–28 August 2016, pp.977–980. New York: IEEE.
17. Liu ZH, Zhang B, Zhu N, et al. Hierarchical network threat situation assessment method for DDoS based on D-S evidence theory. In: *2017 IEEE international conference on intelligence and security informatics (ISI)*, Beijing, China, 22–24 July 2017, pp.49–53. New York: IEEE.
18. Fu YM, Shi YQ, Mu AL, et al. A forecast approach of network security situation base on optimal fuzzy grey. In: *2010 international conference on multimedia communications*, Hong Kong, 7–8 August 2010, pp.218–221. New York: IEEE.
19. Li X, Lu Y and Liu S. Network security situation assessment method based on Markov game model. *TIIS* 2018; 12(5): 2414–2428.
20. Zhang Y, Tan XB and Cui XL. Network security situation awareness method based on Markov game model. *J Softw* 2011; 22(3): 495–508.
21. Li FW and Li QZJ. Improved situation assessment method based on hidden Markov model. *Comput Appl* 2017; 37(5): 1331–1334.
22. Liu S and Liu Y. Network security risk assessment method based on HMM and attack graph model. In: *2016 17th IEEE/ACIS international conference on software engineering, artificial intelligence, networking and parallel/distributed computing (SNPD)*, Shanghai, China, 30 May–1 June 2016, pp.517–522. New York: IEEE.
23. Li XY and Zhao H. Network security situation assessment based on HMM-MPGA. In: *2016 2nd international conference on information management (ICIM)*, London, 7–8 May 2016, pp.57–63. New York: IEEE.
24. Shi LL and Chen J. Assessment model of command information system security situation based on twin support vector machines. In: *2017 international conference on network and information systems for computers (ICNISC)*, Shanghai, China, 14–16 April 2017, pp.135–139. New York: IEEE.
25. Gao YY, Shen YJ, Zhang GD, et al. Information security risk assessment model based on optimized support vector machine with artificial fish swarm algorithm. In: *2015 6th IEEE international conference on software engineering and service science (ICSESS)*, Beijing, China, 23–25 September 2015, pp.599–602. New York: IEEE.
26. Song YQ, Shen YJ, Zhang GD, et al. The information security risk assessment model based on GA¨ C BP. In: *2016 7th IEEE international conference on software engineering and service science (ICSESS)*, Beijing, China, 26–28 August 2016, pp.119–122. New York: IEEE.
27. Li S, Bi F and Chen W. An improved information security risk assessments method for cyber-physical-social computing and networking. *IEEE Access* 2018; 6: 10311–10319.

28. Dong G, Li W and Wang S. The assessment method of network security situation based on improved BP neural network. In: *2018 international conference on computer engineering and networks*, Shanghai, China, 17–19 August 2018, pp.67–76. New York: Springer.
29. Luo B and Liu Y. The risk evaluation model of network information security based on improved BP neural network. In: *2012 international symposium on instrumentation & measurement, sensor network and automation (IMSNA)*, Sanya, China, 25–28 August 2012, pp.189–191. New York: IEEE.
30. Zhang YC, Zhang RC and Liu J. Network security situation assessment using deep self-encoding network. *Comput Eng Appl* 2020; 949(6): 98–104.
31. Badem H, Caliskan A and Basturk A. Classification of human activity by using a stacked autoencoder. In: *2016 medical technologies national congress (TIPTEKNO)*, Antalya, 27–29 October 2016, pp.1–4. New York: IEEE.
32. Verma A and Ranga V. Statistical analysis of CIDDS-001 dataset for network intrusion detection systems using distance-based machine learning. *Proced Comput Sci* 2018; 125: 709–716.