Applied Soft Computing Journal 102 (2021) 107096

Contents lists available at ScienceDirect

Applied Soft Computing Journal

journal homepage: www.elsevier.com/locate/asoc

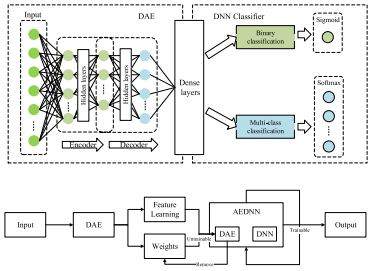
A network security situation assessment method based on adversarial deep learning☆

Hongyu Yang [a], Renyun Zeng [a], Guangquan Xu [b], Liang Zhang [c,*]

[a] School of Computer Science and Technology, Civil Aviation University of China, 300300, Tianjin, China
[b] College of Intelligence and Computing, Tianjin University, 300350 Tianjin, China
[c] School of Information, University of Arizona, AZ 85721 Tucson, USA

GRAPHICAL ABSTRACT

This paper combines Deep Autoencoder (DAE) and Deep Neural Network (DNN) to form a Deep Autoencoder-Deep Neural Network (AEDNN) model to identify different kinds of attacks on the network. For considering the results of feature learning in DAE, this paper establishes an adversarial training process by updating the training weights. Finally use the binary classification results and the multi-class classification results to calculate the quantitative results of the network situation and assess the network security situation.



ARTICLE INFO

Article history:
Received 1 October 2020
Received in revised form 4 December 2020
Accepted 5 January 2021
Available online 11 January 2021

Keywords:
Network security situation assessment
Network attacks
Deep learning
Autoencoder
Adversarial training
Data resampling

ABSTRACT

Aiming at the poor performance and flexibility of traditional assessment methods of network security in dealing with a large number of network attack data, this paper proposes a network security situation assessment method based on adversarial deep learning. Firstly, establish the Deep Autoencoder-Deep Neural Network (AEDNN) model based on Deep Autoencoder (DAE) and Deep Neural Network (DNN). Conduct feature learning on the DAE network and apply the DNN network as a network attacks classifier. In the training process, for considering the results of feature learning in DAE, this paper builds an adversarial training process by changing the training weights. Besides, to increase the performance of the model on the minority network attacks, the Under–Over Sampling Weighted (UOSW) algorithm is designed; Finally, conduct model testing and calculate the network security situation value. The compared results of other models show the proposed model is more accurate for identifying network attacks and can evaluate the network situation more comprehensively and flexibly.

1. Introduction

Network technologies are developing rapidly over the years, at the same time, various attack issues via the Internet occur more frequently, which bring more and more serious damages [1]. The National Internet Emergency Center (CNCERT) reported that, in the first half of 2019, China's network has suffered from a large number of diversified network attacks. Network Security

☆ This work is funded by the Civil Aviation Joint Research Fund Project of National Natural Science Foundation of China under granted number U1833107.

* Corresponding author.
E-mail addresses: hyyang@cauc.edu.cn (H. Yang), zengrenyun@163.com (R. Zeng), losin@tju.edu.cn (G. Xu), liangzh@arizona.edu (L. Zhang).

https://doi.org/10.1016/j.asoc.2021.107096
1568-4946/

Situation Assessment (NSSA) becomes one of the most common and effective solutions, it obtains the real-time situation of network security and provides decision support to network security managers [2].

Situation awareness (SA) is first proposed by Endsley [3], it considers the situation comprehensively and gives a view of its development. In 1999, SA was introduced to the field of network security and first applied to study the next generation of intrusion detection systems [4]. Network situation awareness [5] determines the network situation according to the current environmental factors of the network then to predict its status in the near future. NSSA is an important part of network situation awareness, it finds the potential security risks in the network as soon as possible and fully evaluates the influence degree of these hidden threats, which can help network security managers master the current network situation, to take containment and prevention measures against these threats before the occurrence of network attacks [6].

Great progress has been made in the research of network security situation assessment, however, NSSA does not have a systematic theoretical system so far. Traditional methods based on the mathematical logic model and knowledge reasoning model can integrate the overall situation of the network to a certain extent, and provide decision-making advice for network managers. However, with the network stepping into the era of big data, some traditional methods cannot meet the requirements of dealing with a large number of network traffic and attacks, which usually rely too much on expert evaluations and logical reasoning, so it is difficult to assess the situation according to the real-time state of the network.

Aiming at the shortcomings of the traditional models, this paper introduces a NSSA method based on adversarial deep learning network. To deal with the problems of the extreme imbalance of different types of categories in the dataset, this paper proposes an Under–Over Sampling Weighted (UOSW) algorithm to process the dataset, then Deep Autoencoder (DAE) and Deep Neural Network (DNN) are combined to classify the network attacks. Update the parameters in the adversarial training process to enhance the robustness and the generalization of the network. After obtaining the classification results of network attacks, conduct the impact assessment for each type of attack, and compute the quantitative results of the network security situation. The comparisons with other methods indicate that the method in this paper has better performance for identifying network attacks, especially for the minority network attacks. Besides, it can accomplish the real-time assessment of network security situation with a more efficient and intuitive assessment effect.

The contributions of this paper are summarized as follow:

(1) A deep autoencoder network is improved to detect the network attacks, then the quantitative results of the network security situation are calculated according to the detection results, which is not rely on the experts' evaluations completely as it bases on the network traffic data.
(2) The adversarial training process improves the generalization of the network and a data resample algorithm is proposed to improve the precision of detection of the minority classes.
(3) The experimental results show that compared with other models, the proposed model is more accurate for identifying network attacks especially for the minority attacks, and can evaluate the network situation more comprehensively and flexibly.

2. Related works

NSSA can integrate the overall network environment to provide advice for network managers to reduce the risks caused by network threats, which is a meaningful research topic in the field of network security. Zhao et al. [7] proposed a constructed systematic security assessment model based on the gray relational analysis. Alali et al. [8] took the network attacks as an important indicator of network situation and improved the network situation evaluation model by using fuzzy logic reasoning system. However, these approaches cannot make an immediate response when faced with new types of network attacks.

Neural networks and machine learning are applied successfully in many fields over recent years, experts have begun to try to integrate these technologies into NSSA. An assessment model through an improved Back Propagation (BP) network which is introduced the momentum factor is proposed by Dong et al. [9]. Wen et al. [10] suggested a NSSA method of Naive Bayes classifier to display the overall current network security status dynamically. Hu et al. [11] combined the Support Vector Machine (SVM) and optimized Cuckoo Search (CS) algorithm to predict network security situations. The performance of their method on the KDD dataset achieved high accuracy. The above methods can dynamically assess the network security situation, but when dealing with a large number of network attacks, they have poor efficiency in the real-time assessment.

Deep learning has widespread application in various aspects due to its effectiveness, among which it has an excellent performance in the field of detecting massive network attacks. Hodo et al. [12] showed through experiments that compared with traditional shallow networks methods, deep networks have higher accuracy and effectiveness in detecting network attacks. Althubiti et al. [13] applied Long-Term Memory (LSTM) to train and test the CIDDS-001 dataset. Despite the high accuracy of the experimental results, the homologous training dataset and test dataset were selected in the paper, which did not show the generalization ability of the model. Javaid et al. [14] combined Self-taught Learning (STL) with Sparse Autoencoder (SAE), which greatly improved the detection accuracy of the NSL-KDD dataset. However, there is no comparative analysis of the classification results of each attack in this article, which is prone to have extremely unbalanced test results for attacks that have different sample sizes.

Adversarial Learning is one of the most promising technology in the field of deep learning, it provides security for machine learning in a malicious environment [15]. In the adversarial neural network, it usually adds some noise to the input and trains the network to enhance the robustness of the network. Salem et al. [16] and Hara et al. [17] proposed an improved generative adversarial Network to detect the abnormal traffic in the network while they can only apply in the binary classification.

3. Adversarial deep learning network

Deep learning has been successfully applied in many fields for its strong learning ability, adaptability, and wide coverage. Network attacks in recent years have caused many serious security problems, so apply deep learning to detect network attacks develops into an increasingly meaningful topic [18].

3.1. Deep Autoencoder (DAE)

Autoencoder (AE) is mainly used in data dimension reduction and feature learning and it contains an encoder and decoder. The input data is mapped by the encoder to the decoder so that the actual data can be described with more condensed characteristics. Deep Autoencoder is an improved model of Autoencoder. Hinton

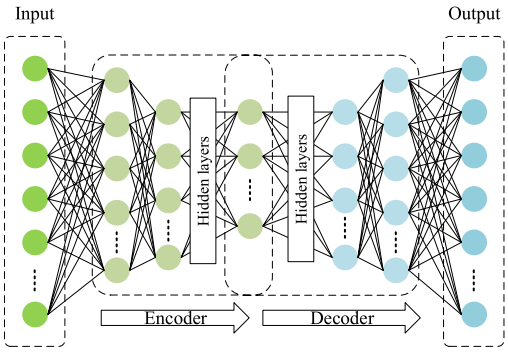


Fig. 1. DAE's network structure.

et al. [19] deepened the network structure of the original Autoencoder and generated the DAE model. DAE's learning ability is improved because of more hidden layers that are more conducive to feature learning. Fig. 1 depicts the network structure of DAE.

The loss function of the DAE is Mean Squared Error (MSE), it calculates the error between the actual data and the predicted data. The formula is shown as follow:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}\left(Y_i - \overline{Y}_i\right)^2 \tag{1}$$

where n is the data size, and Y_i and $\overline{Y}_i$ represent the actual data and predicted data.

3.2. Deep Autoencoder-Deep Neural Network (AEDNN)

Deep Neural Network (DNN) can be considered as a neural network that contains many hidden layers. Because of its accuracy and high efficiency, it is widely used in intrusion detection. Compared with the traditional machine learning classifier, DNN can be used as the classifier to obtain more accurate classification results in less time. Therefore, DNN is selected as the classifier for network attack data. The AEDNN classification model is presented in Fig. 2.

AEDNN model not only can detect normal and abnormal network traffic, but also detect various network attacks. The sigmoid activation function [20] is applied for the binary classification, the value of sigmoid is mapped to an interval of 0 to 1, The closer it is to 1, the more likely it is abnormal network traffic.

When need to detect various network attacks, AEDNN uses the softmax activation function [21]. Softmax maps the output to a range of 0 to 1, the output of softmax is a probability distribution and the sum of each output is 1.

The loss function of the AEDNN model is cross entropy loss function because it minimizes the error between the predicted probability distribution and the actual probability distribution. The cross entropy loss function can be expressed by the following formula:

$$loss = -\frac{1}{n}\sum_{i=1}^{n}\left[Y_i \log \overline{Y}_i + (1 - Y_i)\log\left(1 - \overline{Y}_i\right)\right] \tag{2}$$

where n denotes the total number of samples, Y_i and $\overline{Y}_i$ represent the actual data and predicted data.

3.3. Adversarial training process

To improve the generalization and the robustness [22] of AEDNN, this paper conducts adversarial training by updating the weights of the network. Firstly, train the DAE for feature learning and retain its weight w_1. Then combine DAE and DNN to form AEDNN, and train them together. In the training process, first set the weights of DAE in AEDNN as w_1 and make them untrainable which gets the results of feature learning. After that, make the weights of DAE in AEDNN trainable and train them again, which can take into account the results of feature learning. The adversarial training process is displayed in Fig. 3.

Table 1
NSL-KDD dataset information.

Dataset	Normal	DoS	Probe	R2L	U2R	Total
KDDTrain+	67343	45927	11656	995	52	125973
KDDTest+	9710	7456	2421	2754	202	22543

4. Under–Over Sampling Weighted (UOSW) algorithm

4.1. Dataset description

In this paper, NSL-KDD, a relatively authoritative intrusion dataset in the network security research field is selected as the source dataset for evaluation. The NSL-KDD dataset is enhanced from the KDD99 dataset, its training set contains no redundant or duplicate records, which helps the classifier produce unbiased results [23]. There are 41 features and 5 main attack types in the dataset. This paper uses KDDTrain+ as the train dataset and KDDTest+ as the test dataset, Table 1 presents the information of the NSL-KDD dataset.

4.2. Dataset description

To train the network model more conveniently and accurately, it is necessary to convert the classification features in the source data into digital features and perform numerical normalization.

4.2.1. Feature numericalization

The NSL-KDD dataset has three classification features, are 'protocol_type', 'service', and 'flag', which include 3, 64, and 10 categories, respectively. In this paper, the three classification features are converted into data only represented by 0 and 1 by one-hot encoding. For example, 'protocol_type' has three types of attributes: 'tcp', 'udp' and 'icmp', they will be encoded as (1,0,0), (0,1,0) and (0,0,1) after one-hot encoding. Process the other two features in the same way, the dataset is changed from 41-dimensional features to 116-dimensional features.

4.2.2. Feature normalization

There is a significant difference between the minimum and maximum of the features. To diminish the negative influence of different numerical levels in the model, this paper applies the logarithmic scaling method for scaling feature value, so that they are normalized to the same interval. The formula is as follows:

$$x_{norm} = \frac{(x - x_{min})}{(x_{max} - x_{min})} \tag{3}$$

where x_{max} stands for the maximum and x_{min} denotes a minimum for each feature.

4.3. Under–Over Sampling Weighted (UOSW) algorithm

As depicted from Table 1, the data distribution of the five types of class in the NSL-KDD dataset is very uneven, among which the normal type with the most data size reaches 67 343, while DoS and U2R only contain 52 and 995 piece of data. Extremely unbalanced data distribution will lead to the poor learning effect

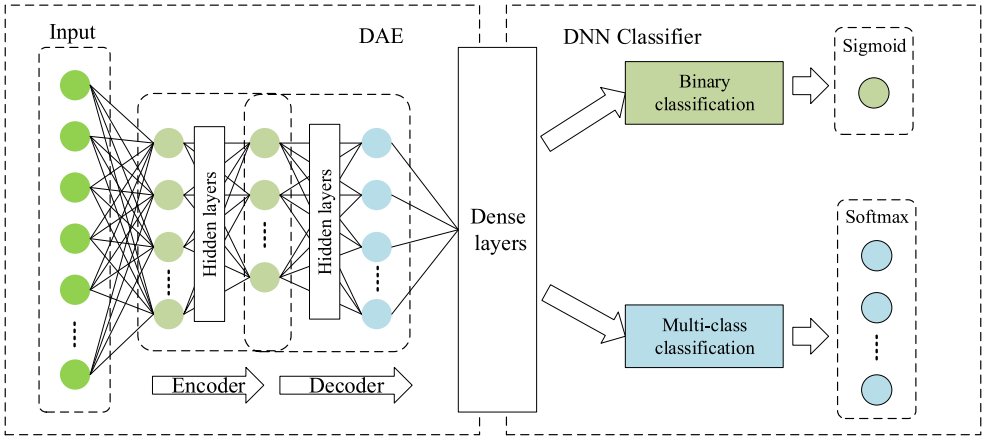


Fig. 2. AEDNN classification model.

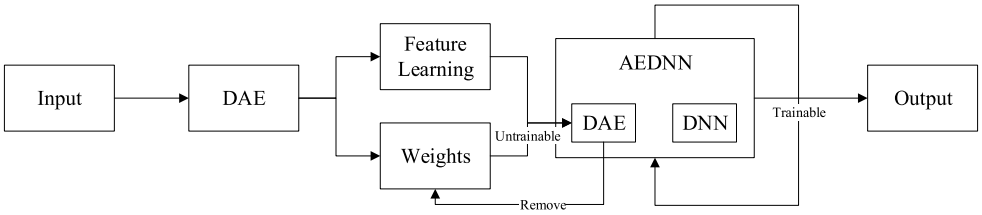


Fig. 3. The adversarial training process.

Table 2
Variables of UOSW algorithm.

Variable	Meaning
n	n classes in the dataset
x_i	original size of each class
w_i	weight of each class in the dataset
U_{i_train}	the training dataset of the majority class i
U_{i_remain}	the remaining dataset of the majority class i
$size_i$	the size of the resampled dataset
$type_i$	each class
Set_{union}	union dataset of resampled majority and minority classes
Set^2	oversampled dataset
Set_{remain}	the remaining part of the majority classes
O_{i_train}	the training set of the minority class

of the model, resulting in that the majority classes with high accuracy and the minority classes with low accuracy.

Oversampling and Undersampling are techniques that are applied to balance the category distribution of a dataset, they are also known as data resampling. Undersampling is applied to decrease the size of majority classes in the imbalanced class distribution. While Oversampling increases the data size of minority samples in the data to achieve data balance.

This paper proposes an Under–Over Sampling Weighted (UOSW) algorithm which is combined with over-sampling, undersampling, and weighted to deal with the disadvantages of imbalanced data and increase the accuracy in detecting the minority classes. The algorithm is divided into the coming steps:

The variables and the meanings of them of the algorithm are shown in Table 2.

Step 1. Calculate the weight of each class in the dataset. In the network training process, if the data size of each class in the training set is very approximate (reaches average), the model which learns from the training set will have high accuracy. Therefore, this paper measures the difference between the original sample size of each class and the ideal sample size as the weight to achieve balance for each class. Suppose the dataset contains n classes and each class has x_i samples. Their weight w_i in the dataset can be expressed by the coming formula:

$$w_i = \frac{\sum_{i=1}^{n} x_i}{x_i \times n} \tag{4}$$

Step 2. Data undersampling. Conduct data undersampling to the majority classes and change the data size of them. This paper applies the 'train_test_split' method of sklearn in Python to divide each majority class into U_{i_train}, U_{i_remain}. Add U_{i_train} to Set^2 which is the processed training set, and add U_{i_remain} to Set_{remain} which is used for oversampling the minority classes, where the size of U_{i_train} is $size_i = x_i \times w_i$.

Step 3. Data oversampling. This paper uses the oversampling algorithm SMOTE [24] to process the minority class samples, the core of SMOTE is generating the new minority class samples based on the existing minority class samples. Since the SMOTE algorithm is originally aimed at the problem of binary classification, the algorithm is improved for the multi-class problems as follows:

The class needs to be oversampled is denoted as $type_i$, n denotes there are n categories, and its original dataset and data size are Set_i, x_i.

(1) Combine the data of other categories. The other categories contain the majority classes and the minority classes, and the majority classes are processed in Step 2. Union the remaining part of the majority classes Set_{remain} and the original datasets of the minority classes to Set_{union}.
(2) Change labels. Change the labels of Set_{union} to the same type but different from $type_i$.
(3) Define data size. To balance the dataset, the minority class samples need to be expanded, and the expanded data size is $size_i$, where $size_i = x_i \times w_i$, and w_i is the weight of $type_i$.
(4) Data oversampling. Apply the SMOTE method of imblearn library in Python, combine with Set_{union} to generate the required data, and denote the oversampled data as O_{i_train}.

(5) Repeat (1)–(4). Oversample all classes whose data size is less than the average size. Finally, combine all the datasets of undersampling classes and oversampling classes to form a training set for multi-classification.

Algorithm 1 displays the UOSW algorithm.

Algorithm 1: UOSW algorithm

Data: original train dataset Set^1, initial size of Set^1 $total$, initial number of classes of n, initial dataset of each class Set_i, initial size of Set_i x_i

Result: resampled dataset Set^2

```
average←total/n;
w_i←total/(x_i × n); list w=[w_1, w_2 .., w_n];
size_i←x_i × w_i; list size=[size_1, size_2 .., size_n];
Set^2←null; Set_remain←null; i, j←0, 0;
while i < n do
  if x_i > average then
    U_i_train, U_i_remain← Data undersampling (Set_i, size_i);
    Set^2←Set^2+U_i_train;
    Set_remain←Set_remain+ U_i_remain;
  end
  if x_i == average then
    Set^2←Set^2+Set_i;
  end
end
while j < n do
  if x_i < average then
    O_i_train← Data oversampling (Set_i, size_i, Set_remain);
    Set^2←Set^2+O_i_train;
  end
end
```

5. Proposed NSSA model

The NSSA model designed in this paper includes three sections: situation acquisition, situation analysis, and situation assessment. The structure of the proposed NSSA model is shown in Fig. 4.

As can be seen in Fig. 4, there are three parts in this model:

(1) Situation acquisition. In this stage, we collect the traffic data in the network. To simulate the situation that the network is dealing with massive traffic data, this paper selects the NSL-KDD dataset mentioned above as the network traffic. After preprocessing the data, input the data to the AEDNN model for training.
(2) Situation analysis. Input the test dataset into the trained model, and record the output binary classification results and multi-classification results for the calculation of network security situation value.
(3) Situation assessment. Compute the probability of attacks and the impact of each kind of attack based on the results of test classification. Besides, network security situation values are calculated and evaluated.

5.1. Attack probability

The attack probability p is based on the binary classification results of the AEDNN network. If a piece of network traffic data is classified as abnormal network data, it counts 1. Otherwise, it counts 0. Define n as the total network traffic over a period of time and the attack probability p is calculated by the following formula:

$$p = \frac{\sum_{i=1}^{n} AEDNN\ (i)}{n} \tag{5}$$

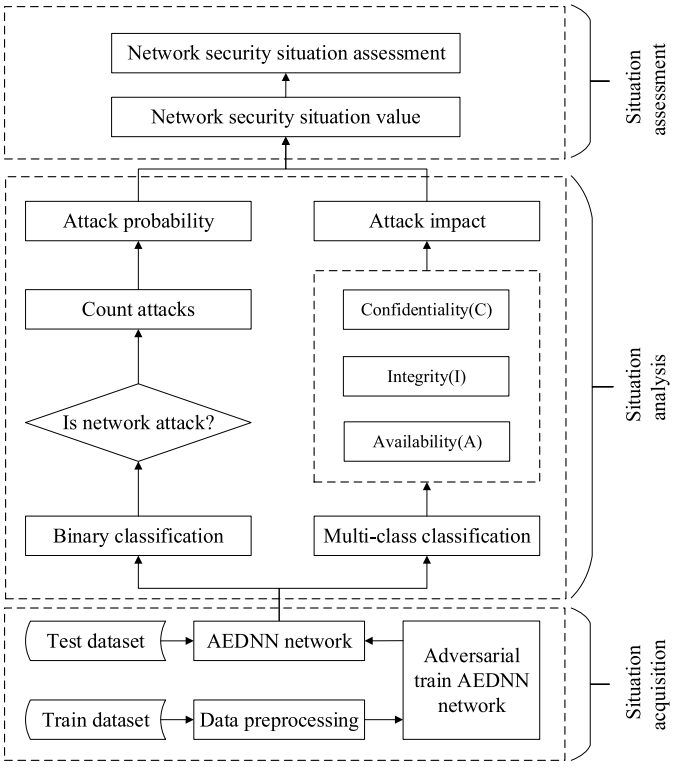


Fig. 4. The structure of the proposed NSSA model.

Table 3
Impact scores of C, I and A.

Indicator	Impact	Score
Confidentiality (C)	None (N)/Low (L)/High (H)	0/0.22/0.56
Integrity (I)	None (N)/Low (L)/High (H)	0/0.22/0.56
Availability (A)	None (N)/Low (L)/High (H)	0/0.22/0.56

5.2. Attack impact

NSL-KDD dataset includes five types of attacks: Normal, DoS, U2R, R2L, and Probe [25].

(a) Denial of Service Attack (DoS)
This attack sends a large amount of traffic to computers of target users or network servers, which makes the intended users inaccessible.
(b) User to Root Attack (U2R)
This attack tries to get a root account and superuser privilege.
(c) Remote to Local Attack (R2L)
This kind of attack enables the intruders to gain access to the local machine which they do not originally have an account for this.
(d) Probe Attack
This attack collects the information of the network which is required before initiating other attacks.
(e) Normal
Normal network data traffic.

This paper has formulated an attack impact rating scale according to the common vulnerability scoring system (CVSS) [26,27]. The scores of confidentiality (C), integrity (I), and availability (A) are described in Table 3.

Table 4
Network security situation severity levels.

V	*NSSL*
0.00∼0.20	safety
0.21∼0.40	low-risk
0.41∼0.60	moderate-risk
0.61∼0.80	high-risk
0.81∼1.00	super-risk

The impact score of each attack is calculated by the following formula:

$$I_i = C_i + I_i + A_i \tag{6}$$

where i denotes the type of attacks.

5.3. Network security situation quantification

Quantify the network security situation can analyze the overall network situation more intuitively. The quantitative assessment process of the proposed method mainly includes four steps: attack analysis, calculate the impact of attacks, compute quantitative results of NSSA, and assess the quantitative network security situation. Detailed processing steps are as follows:

Step 1. Attacks analysis. We randomly select several groups of data from the test dataset and input them into the AEDNN model than perform binary and multi-classification on them. The proportion of attacks detected in the binary classification is recorded as the attack probability p.

Step 2. Calculate the impact of attacks. According to formula 6, we calculate the impact scores of attacks.

Step 3. Compute the quantitative results of NSSA. The network security situation value V can be obtained by formula 7:

$$V = \frac{p \times \sum_{i=1}^{n} I_i \times t_i}{N - t_{nml}} \tag{7}$$

where n stands for the types of attacks and t_i indicates each attack detected times. Because the attack impact score of type normal is 0, the total number of network traffic N should exclude the number of the normal type which denotes as t_{nml}.

Step 4. Assess the quantitative network security situation. This paper develops the classification of the network security situation and divides it into five levels which are shown in Table 4. In Table 4, V denotes the network security situation value and *NSSL* represents the corresponding network security situation levels.

6. Experiment results

6.1. Evaluation metrics

This paper uses the metrics defined below:

True Positive (*TP*): it stands for the number of attacks that are classified as attacks;

False Positive (*FP*): it stands for the number of normal data that is identified as an attack;

True Negative (*TN*): it means that the number of normal data is e identified as normal;

False Negative (*FN*): it means that the number of attacks that are classified as normal.

Confusion Matrix: the confusion matrix can reflect the performance of the machine learning model directly. The rows and columns of the matrix denote the predicted classes and the actual classes [28]. *Precision*, *Recall,* and other evaluation metrics are calculated according to the confusion matrix.

Precision: it indicates the attacks predicted by the model are the actual attacks. The higher Precision is, the lower the false alarm is.

$$precision = \frac{TP}{TP + FP} \tag{8}$$

Recall: it refers to the percentage of correctly classified attacks to the number of actual attacks.

$$recall = \frac{TP}{TP + FN} \tag{9}$$

F-score: it takes into account both precision and recall:

$$F - score = 2 \times \frac{precision \times recall}{precision + recall} \tag{10}$$

Receiver Operating Characteristic (ROC) curve is a performance measurement for classification problems at various thresholds settings [29]. The y-axis of the ROC curve is the True Positive Rate (*TPR*) and the x-axis is the False Positive Rate (*FPR*). Area Under Characteristic (AUC) is the area under the ROC curve. The higher the AUC is, the better the model is.

$$TPR = \frac{TP}{TP + FN} \tag{11}$$

$$FPR = \frac{FP}{FP + TN} \tag{12}$$

6.2. Binary classification results

In the task of binary classification, the model only needs to distinguish whether traffic data is an attack or normal data. To examine the effectiveness of the model, this paper also compares it with DT [12], SVM [12], and LSTM [13].

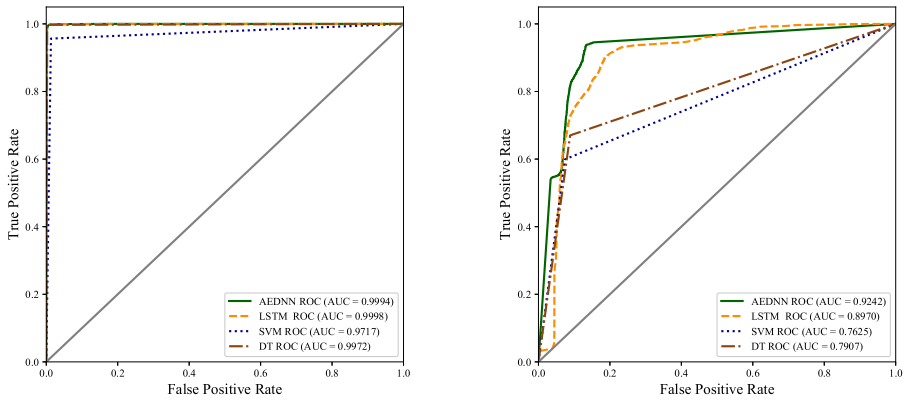
As shown in Table 1, the data amount of normal type in the training dataset is not much different from that of the sum of all other attack types, it is not necessary to apply the UOSW algorithm to process the data. Eighty percent of the data in KDDTrain+ is used for training, the remaining 20% of KDDTrain+ and KDDTest+ are tested respectively. The binary classification results of the four models are shown in Fig. 5.

Fig. 5(a) shows that, when the four models are tested with the 20% of KDDTrain+ dataset, they all show good accuracy and performance. This is because the test dataset and train dataset come from the same dataset, and their data structure is the same, so the models will get ideal results.

However, we found in Fig. 5(b) that if we use the KDDTest+ dataset as a test dataset, the accuracy of four network models is decreased. Because there are some data with a different structure from the training dataset. This situation is very similar to the real network situation that we will face many attacks the model never learned. Fig. 5(b) shows that AEDNN is significantly better than the other three models in this case, and its accuracy is nearly 13.35%, 16.17%, and 2.72% higher than DT, SVM, and LSTM respectively, which indicates that the AEDNN model has good generalization.

6.3. Multi-class classification results

This paper applies the KDDTest+ dataset to test the five models and select precision, recall, and F-score as the evaluation metrics to compare and analyze various models. Precision and recall are calculated according to the confusion matrix. Each row of the matrix represents the predicted class and each column represents the actual class, which means that the diagonal of the matrix denotes the number of the correct classification of each class.



(a) 20% of KDDTrain+ (b) KDDTest+

Fig. 5. The binary classification results of four models.

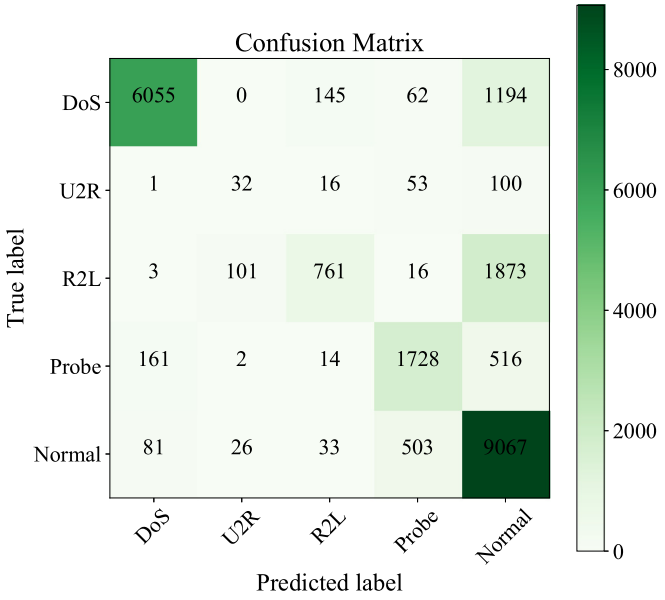


Fig. 6. Confusion matrix of the AEDNN+UOSW.

Table 5
Precisions (%) of five models.

Model	DoS	U2R	R2L	Probe	Normal	Total
DT	82.64	1.98	22.33	52.91	93.06	76.08
SVM	75.34	0.00	0.00	61.21	92.91	72.26
LSTM	79.29	2.97	3.05	68.19	92.46	73.77
AEDNN	77.44	19.31	28.18	73.81	91.80	76.70
AEDNN+UOSW	81.21	15.84	27.63	71.38	93.38	78.26

The confusion matrix of the AEDNN applied UOSW algorithm is shown in Fig. 6.

As can be seen from Fig. 6, the majority classes reach a high accuracy and the minority classes have low accuracy. The training of deep learning models is a self-learn process, which means that the model can learn from the data itself then present the features of data. The model can learn more from the majority size of data, so the model performs better in the majority classes.

The precision, recall, and F-score of each class of the five models are depicted in Tables 5–7.

It can be seen from the tables that in the SVM and LSTM model, the detection rate of 'U2R' and 'R2L' is relatively low. Due to the lack of available training data, these models are not

Table 6
Recall (%) of five models.

Model	DoS	U2R	R2L	Probe	Normal	Total
DT	91.63	30.77	77.55	56.91	70.81	76.67
SVM	93.21	0.00	0.00	70.61	62.57	65.36
LSTM	89.37	4.55	70.00	80.81	65.85	74.11
AEDNN	87.71	20.74	61.15	72.23	74.10	76.34
AEDNN+UOSW	96.10	19.88	78.53	73.16	71.11	80.04

Table 7
F-scores (%) of five models.

Model	F-score
DT	76.37
SVM	68.64
LSTM	73.94
AEDNN	76.82
AEDNN+UOSW	79.14

sensitive to these two kinds of attacks. Compared with the DT network, it achieves better accuracy and recall rate in 'U2R' and 'R2L' attacks, but its 'Probe' classification result is not as good as the two networks mentioned above.

The performance analysis shows that the proposed model is superior to other models in high precision, recall, and F-score. The simulation results demonstrate that the AEDNN model increases the accuracy of the detection of the minority classes, and does not reduce the performance of the model to detect the majority classes.

Besides, the performance of the AEDNN network is more stable after combining our UOSW algorithm. Compared with DT, SVM, and LSTM models, the proposed model improves by 2.77%, 10.5%, and 5.2% respectively on F-score.

6.4. Quantification results of NSSA

In our method, we randomly select the same number of test samples from the test dataset and carry out 50 groups of network attack test experiments. The network security situation values of the tests are shown in Fig. 7.

Fig. 7 shows that the network security situation values calculated based on the AEDNN model are closest to the actual security situation values of the samples. Compared with the two traditional machine learning models, SVM and DT, the two deep learning methods, AEDNN and LSTM can better represent the real situation of the data.

The reason why the situation values of the other three models are quite different from the actual situation values is that there

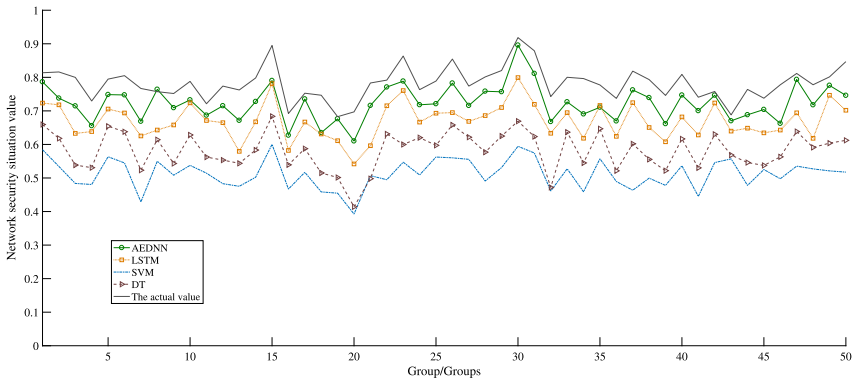


Fig. 7. Network security situation values.

are some attack types with a small number of training samples, which makes the models unable to learn the characteristics of such attacks comprehensively.

UOSW algorithm is applied in the AEDNN model, which improves the accuracy of detecting the minority classes. There are some differences between the situation values calculated by the proposed model and the actual situation values, but most of the network security situation values belong to the same network security severity class.

7. Conclusion

To address the shortcomings of the traditional methods of NSSA that have low efficiency when deal with a large number of network data, this paper proposes a network security situation assessment method based on adversarial deep learning. We first combine the DAE and DNN to establish the AEDNN network, then conduct adversarial network training which improves the generalization and the robustness of the network. The experimental results demonstrate that the proposed method is superior to other models in the task of binary classification. Besides, this model combines with the proposed UOSW algorithm and get higher accuracy in detecting the minority classes. Therefore, the model can measure the network security situation more accurately and evaluate it more comprehensively.

The future work is to test our method in other datasets to verify the validity and generality of the model. Besides, we are going to optimize the calculation method of network security situation value and divide the levels of it in more detail.

CRediT authorship contribution statement

Hongyu Yang: Conceptualization, Methodology. **Renyun Zeng:** Software, Writing - original draft preparation. **Guangquan Xu:** Writing - reviewing and editing. **Liang Zhang:** Supervision, Validation.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment

This work was supported by the Civil Aviation Joint Research Fund Project of the National Natural Science Foundation of China under Grant no. U1833107.

References

[1] S. Rathore, P.K. Sharma, V. Loia, Y.-S. Jeong, J.H. Park, Social network security: Issues, challenges, threats, and solutions, Inf. Sci. 421 (2017) 43–69.
[2] Y.-B. Leau, S. Manickam, Y.-W. Chong, Network security situation assessment: A review and discussion, in: Information Science and Applications, Springer, 2015, pp. 407–414.
[3] M.R. Endsley, Design and evaluation for situation awareness enhancement, in: Proceedings of the Human Factors Society Annual Meeting, Vol. 32, SAGE Publications Sage CA, Los Angeles, CA, 1988, pp. 97–101.
[4] T. Bass, et al., Multisensor data fusion for next generation distributed intrusion detection systems, in: Proceedings of the IRIS National Symposium on Sensor and Data Fusion, Vol. 24, Citeseer, 1999, pp. 24–27.
[5] G.P. Tadda, J.S. Salerno, Overview of cyber situation awareness, in: Cyber Situational Awareness, Springer, 2010, pp. 15–35.
[6] X. Cheng, S. Lang, Research on network security situation assessment and prediction, in: 2012 Fourth International Conference on Computational and Information Sciences, IEEE, 2012, pp. 864–867.
[7] X. Zhao, H. Xu, T. Wang, X. Jiang, J. Zhao, Research on multidimensional system security assessment based on ahp and gray correlation, in: Chinese Conference on Trusted Computing and Information Security, Springer, 2019, pp. 177–192.
[8] M. Alali, A. Almogren, M.M. Hassan, I.A. Rassan, M.Z.A. Bhuiyan, Improving risk assessment model of cyber security using fuzzy logic inference system, Comput. Secur. 74 (2018) 323–339.
[9] G. Dong, W. Li, S. Wang, X. Zhang, J. Lu, X. Li, The assessment method of network security situation based on improved bp neural network, in: International Conference on Computer Engineering and Networks, Springer, 2018, pp. 67–76.
[10] Z. Wen, C. Cao, H. Zhou, Network security situation assessment method based on naive bayes classifier, J. Comput. Appl. (8) (2015) 12.
[11] J. Hu, D. Ma, C. Liu, Z. Shi, H. Yan, C. Hu, Network security situation prediction based on mr-svm, IEEE Access 7 (2019) 130937–130945.
[12] E. Hodo, X. Bellekens, A. Hamilton, C. Tachtatzis, R. Atkinson, Shallow and deep networks intrusion detection system: A taxonomy and survey, 2017, arXiv preprint arXiv:1701.02145.
[13] S.A. Althubiti, E.M. Jones, K. Roy, Lstm for anomaly-based network intrusion detection, in: 2018 28th International Telecommunication Networks and Applications Conference (ITNAC), IEEE, 2018, pp. 1–3.
[14] A. Javaid, Q. Niyaz, W. Sun, M. Alam, A deep learning approach for network intrusion detection system, in: Proceedings of the 9th EAI International Conference on Bio-inspired Information and Communications Technologies (formerly BIONETICS), 2016, pp. 21–26.
[15] A. Kurakin, I. Goodfellow, S. Bengio, Adversarial machine learning at scale, 2016, arXiv preprint arXiv:1611.01236.
[16] M. Salem, S. Taheri, J.S. Yuan, Anomaly generation using generative adversarial networks in host-based intrusion detection, in: 2018 9th IEEE Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON), IEEE, 2018, pp. 683–687.
[17] K. Hara, K. Shiomoto, Intrusion detection system using semisupervised learning with adversarial auto-encoder, in: NOMS 2020-2020 IEEE/IFIP Network Operations and Management Symposium, IEEE, 2020, pp. 1–8.
[18] A. Aldweesh, A. Derhab, A.Z. Emam, Deep learning approaches for anomaly-based intrusion detection systems: A survey, taxonomy, and open issues, Knowl.-Based Syst. 189 (2020) 105124.
[19] G.E. Hinton, R.R. Salakhutdinov, Reducing the dimensionality of data with neural networks, Science 313 (5786) (2006) 504–507.
[20] J. Han, C. Moraga, The influence of the sigmoid function parameters on the speed of backpropagation learning, in: International Workshop on Artificial Neural Networks, Springer, 1995, pp. 195–201.

[21] I. Goodfellow, Y. Bengio, A. Courville, 6.2. 2.3 softmax units for multinoulli output distributions, in: Deep Learning, MIT Press, 2016, pp. 180–184.
[22] Y. Vorobeychik, M. Kantarcioglu, Adversarial machine learning, Synth. Lect. Artif. Intell. Mach. Learn. 12 (3) (2018) 1–169.
[23] R. Bala, R. Nagpal, A review on kdd cup99 and nsl-kdd dataset, Int. J. Adv. Res. Comput. Sci. 10 (2) (2019) 64.
[24] N.V. Chawla, K.W. Bowyer, L.O. Hall, W.P. Kegelmeyer, Smote: synthetic minority over-sampling technique, J. Artif. Intell. Res. 16 (2002) 321–357.
[25] D.D. Protić, Review of kdd cup'99, nsl-kdd and kyoto 2006+ datasets, Vojnotehnički glasnik 66 (3) (2018) 580–596.
[26] C. Team, Common vulnerability scoring system v3. 0: Specification document, [EB/OL], 2015, https://www.first.org/cvss/v3.0/specification-document/ Accessed September 9, 2020.
[27] E. Doynikova, A. Chechulin, I. Kotenko, Analytical attack modeling and security assessment based on the common vulnerability scoring system, in: 2017 20th Conference of Open Innovations Association (FRUCT), IEEE, 2017, pp. 53–61.
[28] D.M. Powers, Evaluation: from precision, recall and f-measure to roc, informedness, markedness and correlation, 2011.
[29] R.S. Peres, J. Barata, P. Leitao, G. Garcia, Multistage quality control using machine learning in the automotive industry, IEEE Access 7 (2019) 79908–79916.