Computer Modeling in Engineering & Sciences

Tech Science Press

DOI: 10.32604/cmes.2021.016866

ARTICLE

A Step-Based Deep Learning Approach for Network Intrusion Detection

Yanyan Zhang[1] **and Xiangjin Ran**[2,*]

[1]Jilin Business and Technology College, Changchun, 130507, China

[2]College of Earth Sciences, Jilin University, Changchun, 130061, China

[*]Corresponding Author: Xiangjin Ran. Email: ranxiangjin@jlu.edu.cn

Received: 04 April 2021 Accepted: 27 May 2021

ABSTRACT

In the network security field, the network intrusion detection system (NIDS) is considered one of the critical issues in the detection accuracy and missed detection rate. In this paper, a method of two-step network intrusion detection on the basis of GoogLeNet Inception and deep convolutional neural networks (CNNs) models is proposed. The proposed method used the GoogLeNet Inception model to identify the network packets' binary problem. Subsequently, the characteristics of the packets' raw data and the traffic features are extracted. The CNNs model is also used to identify the multiclass intrusions by the network packets' features. In the experimental results, the proposed method shows an improvement in the identification accuracy, where it achieves up to 99.63%. In addition, the missed detection rate is reduced to be 0.1%. The results prove the high performance of the proposed method in enhancing the NIDS's reliability.

KEYWORDS

Network intrusion detection system; deep convolutional neural networks; GoogLeNet Inception model; step-based intrusion detection

1 Introduction

With the emergence of developing computer technology, networks' users and interconnected devices and data connected to the Internet are rapidly increased [1]. Therefore, the opportunity to affect these users, devices, and data by hackers and illegal networks is also increased [2]. Hackers are usually hiding behind the big data traffic, and intrude the key infrastructure of critical systems, such as banks, power systems, online shopping malls, and government agencies, to perform illegal activities [3,4]. Accordingly, the intrusion detection system (IDS) is proposed as an active defense technology to quickly and accurately detect unauthorized connections [4–6].

Recently, several rules were used by IDSs to detect the targeted networks' intrusion, which proved its efficiency [7]. In contrast, these systems have some problems, such as detecting new intrusions, relying heavily on engineers' skill, low accuracy of intrusion detection, and high false-positive and false-negative rate. In addition, in the case of big data, the IDSs cannot meet the needs of fast, stable and efficient detection of intrusion data from massive network connections [8]. With the emergence of machine learning, the IDSs are significantly improved by

enhancing the detection accuracy and efficiency [9–11]. However, traditional machine learning methods are still ineffective to meet massive data and high computation needs.

In 2006, deep learning technologies [12] were proposed and successfully applied in many fields, particularly, recognition and detection algorithms. Deep learning uses convolution, pooling and other neural network layers to build different models. Deep learning can iteratively train data to extract its characteristics to avoid human intervention. Recently, lots of deep learning models are successfully applied for image classification [13], speech recognition [14,15], medical diagnosis [16], rock recognition [17], intelligent transportation [18] and other fields [4,19].

In intrusion detection research, various intrusion detection models have been proposed on the basis of deep learning technologies [1,12,20,21]. As unsupervised deep learning model, deep encoder (DE) [22–24], support vector machine (SVM) [4], recurrent neural network (RNN) [20,25], transfer learning [26], and reinforcement learning [27] have been used in the detection of network intrusions. Most of these models can detect new types of attacks, significantly improve intrusion detection accuracy, and reduce the rate of false-negative and false-positives. Therefore, the IDSs on the basis of deep learning receive extensive research and attention [28–30].

Deep learning models provide several advantages to solve different problems optimally [31–36]. For instance, the RNN model can memorize and identify the previous features [20], and the Auto Encoder network can automatically learn the sample features [12]. In addition, several studies have proposed deep learning hybrid models intrusion detection to maximize the advantages of the used models [10]. For instance, Zhang et al. [37] proposed a hybrid intrusion detection model on the basis of Deep Belief Network (DBN) and Twin Support Vector Machine (TSVM), and Wang et al. [38] proposed an intrusion detection model on the basis of Convolutional Neural Network-Support Vector Machine (CNN-NSVM). Aljawarneh et al. [10] proposed a new hybrid model that combined some classifiers to classify the binary and multiclass by selecting the most critical features. The results shown by the hybrid model proved its significant training time reduction and accuracy of detection improvement.

NSL-KDD [39] dataset (i.e., the upgrade version of KDD99 [40] dataset) is often used as a benchmark to evaluate IDS. Most of the current research used NSL-KDD dataset in the evaluation processes. However, even network environments are rapid growth, the datasets are still old, and the network intrusion behavior is often hidden in the network huge data flow. The network traffic characteristics consider only one-sided for intrusion detection, which makes it ineffective for detection. In order to address such issue, Liang et al. [41] used a one-hot encoding method to encode the original network packets using the UNSW-NB15 [42] dataset. In addition, the proposed method discarded the empty packets containing the data header without actual packet content, and formed two-dimensional data. Subsequently, the authors used the GoogLeNet concept model for training, avoiding the shortcomings of using public datasets, and achieving better detection results. However, some attacks, such as Synchronize Sequence Numbers (SYN) Flood, may escape IDS using such methods because these attack packets often do not carry application-layer data.

In this paper, a step-based deep learning intrusion detection method is proposed to improve the accuracy of intrusion detection and reduce the rate of the missing report. In terms of the network data flow, the method first judges whether the data packet contains application-layer data. In terms of the packet containing application-layer data, the original data packet is detected using the GoogLeNet Inception intrusion model on the basis of the original traffic to achieve the purpose of rapid detection. In addition, the feature-based CNNs model is used to identify the

type of packets without application-layer data to avoid missing intrusion connection and improve the accuracy of intrusion detection. Subsequently, the proposed method is implemented on the basis of TensorFlow-GPU framework. Finally, the model is applied in a laboratory environment.

The remainder of this paper is organized as follows. Section 2 presents the related works, particularly the deep learning approaches. Section 3 describes the benchmark datasets and step-based deep learning approach for intrusion detection. Factor analysis that affects the identification accuracy, such as the type of model, sample size, and model level is presented in Section 4. In addition, the evaluation results are presented in this section. Section 5 provides the conclusion of this research.

2 Related Work

NIDS is playing an essential role in the network security field. According to the NIDS's position in the network, the efficient IDS must contain a series of indicators, such as high accuracy, low false detection rate, low miss detection rate, and low delay. With the development of artificial intelligence technology, many deep learning models [8,13,21–23,25] and optimization algorithms [43–50] have been proposed. The accuracy of classification and object detection using computers is getting higher and higher. A massive number of studies have been proposed starting from the old methods using rule-based [7] and feature-based [10,19,42,51,52] intrusion detection to the modern deep learning methods [1–5,21,23–25].

The intrusion detection methods that used machine learning models, such as random forest (RF), SVM [19] and decision tree [11], attracted the researchers' attention, where these methods superior intrusion detection technology on the basis of rule base [7]. A large number of anomaly classification algorithms have been proposed. Assiri [9] proposed a method for anomaly classification of network attack detection using genetic algorithm-based RF. Aljawarneh et al. [10] used feature selection analysis to build a hybrid efficient model for anomaly-based intrusion detection. Sivatha et al. [11] construct a light weight IDS based on decision tree to detect anomalies in networks. Ambusaidi et al built an IDS using a feature selection algorithm based on filter [42]. These methods could identify the anomaly intrusion, but were difficult in feature selection.

Recently, the researchers' attention became more interested in using deep learning methods instead of the early rule-based and machine learning methods, particularly in IDS. Deep learning methods can be easily applied to IDS to automatically learn the characteristics of the existing data through training and obtain the parameters to identify the unknown data. Several intrusion detection (anomaly detection) methods using deep learning methods, such as deep convolutional neural networks (DCNNs), GoogLeNet, RNN [25], long short-term memory (LSTM) [36] and DBN, are proposed and applied for IDS using different scenarios. In the evaluation results, the proposed methods proved their robust performance in the detection. In addition, some hybrid intrusion detection models have also been proposed. Zhang et al. [37] proposed a hybrid intrusion detection model based on DBN and TSVM. Wang et al. implemented an intrusion detection model based on CNN and NSVM [38].

3 Data Processing and Methodology

In this section, the datasets used for training, testing, and evaluating are introduced. Also, the methodology of the step-based deep learning for network intrusion detection is illustrated.

3.1 Datasets and Processing

(1) NSL-KDD dataset

Several intrusion detection methods used KDD99 and NSL-KDD datasets for training and evaluating the methods. NSL-KDD is created using KDD99 dataset by integrating many network data traffic characteristics, including normal connection and four types of attack connection, as shown in Tab. 1. It is more suitable for training the classification of normal data and attack data without application layer data.

Table 1: Data distribution in NSL-KDD dataset

Connection type		Record count
Normal		77054
Abnormal	DoS	51668
	Probe	12762
	R2L	3194
	U2R	119

Accordingly, NSL-KDD dataset is used in the evaluation processes of this research. Each record in the dataset refers to a network connection which contains 41 features. Given that the primary objective of this study is related to the network intrusion, the features related to hosting intrusion are not considered. These features are removed and ignored; therefore, only 28 features are used, including three features are characters, which are protocol, service and flag. During training process, these three characters' features can't be used for numerical calculation, so they are coded by the one-hot encoding [39]. There are 3 protocol types in the protocol feature, while 70 service types and 11 network connection states. After one-hot encoding, each sample creates 109 items and fills 12 zero values to create a two-dimensional matrix with a size of 11×11.

(2) UNSW-NB15 dataset

Australian Centre Cyber Security created UNSW-NB15 dataset on the basis of the IXIA Perfect Storm tool [53,54]. This dataset provides up to 100 GB of raw data traffic, including nine types of attacks: Fuzzers, Analysis, Backdoors, DoS, Exploits, Generic, Reconnaissance, Shellcode, and Worms (Tab. 2). UNSW-NB15 dataset is used on the basis of a large number of original data samples to train the classification of network data packets that contain application-layer data.

UNSW-NB15 dataset contains a massive amount of data with the same MAC address and IP address that may affect the accuracy of detection. Therefore, the Ethernet header and IP header in the packet should be removed to eliminate the influence of fixed MAC address and IP address on the model's accuracy and get the simplified packet (Reduced-Data) [39]. When finished reducing, the rest data contains 1460 bytes content maximum, forms a two-dimensional matrix with a size of 38×38 by removing the last bytes if more than 1444 bytes or padding zeros to 1444 bytes if less than 1444 bytes. Subsequently, the Reduced-Data is standardized and normalized, and then send to the deep learning model as input data for training or inference.

3.2 Methodology

The general architecture of the step-based deep learning intrusion detection system (SDL-IDS) can be shown in Fig. 1. The SDL-IDS contains two modules, including the Raw-Data-based deep

learning intrusion detection module (RDDL-IDM) and the Feature-based deep learning intrusion detection module (FDL-IDM).

Table 2: Data distribution in UNSW-NB15 dataset

Connection type		Record count
Normal		2,218,761
Attack	Fuzzers	24,246
	Analysis	2,677
	Backdoors	2,329
	DoS	16,353
	Exploits	44,525
	Generic	215,481
	Reconnaissance	13,987
	Shellcode	1,511
	Worms	174

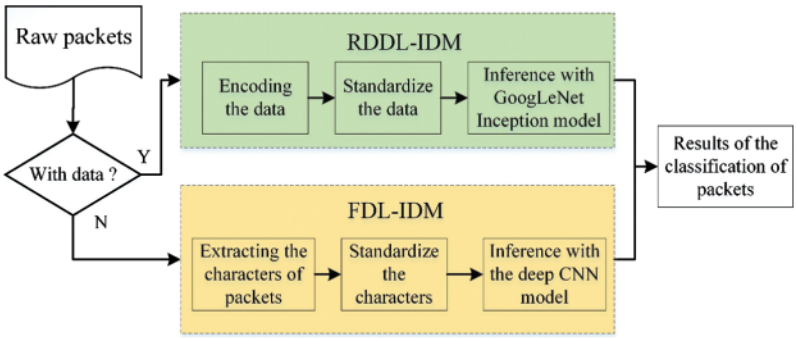

Figure 1: The general architecture of SDL-IDS

(1) RDDL-IDM

RDDL-IDM detects network attacks on the original network packets with application-layer data to avoid the tedious steps of feature extraction and improves the detection efficiency. Subsequently, it is classified by GoogLeNet model to obtain the detection results.

GoogLeNet model outperformed all compared models in the ILSVRC competition in 2014 [55]. GoogLeNet model contains 22 layers. Its most prominent feature is that the final fully connected layer is replaced by the global average pooling layer, making the model training faster and reducing over-fitting. The concept module is considered the most significant module of GoogLeNet model, where it solves the problems of over-fitting and gradient dispersion by increasing the depth and width of the network and reducing the parameters at the same time.

The concept V1 model contains four classes, as shown in Fig. 2b. The first class convolutes the input by 1×1, which is an important structure proposed in Network. 1×1 convolution filter can organize information across channels and improve the network's expression ability. At the

same time, the dimension of the output channel can be increased or decreased. 1 × 1 convolution filter is also used in the second class, and 3 × 3 convolution operation equivalent to two feature transformations is connected. The third class is similar to the second class, which use 1 × 1 and 5 × 5 convolution operations. In the fourth class, 1 × 1 convolution operation is used after applying a max-pooling function with 3 × 3. These four classes are aggregated using the concatenate function.

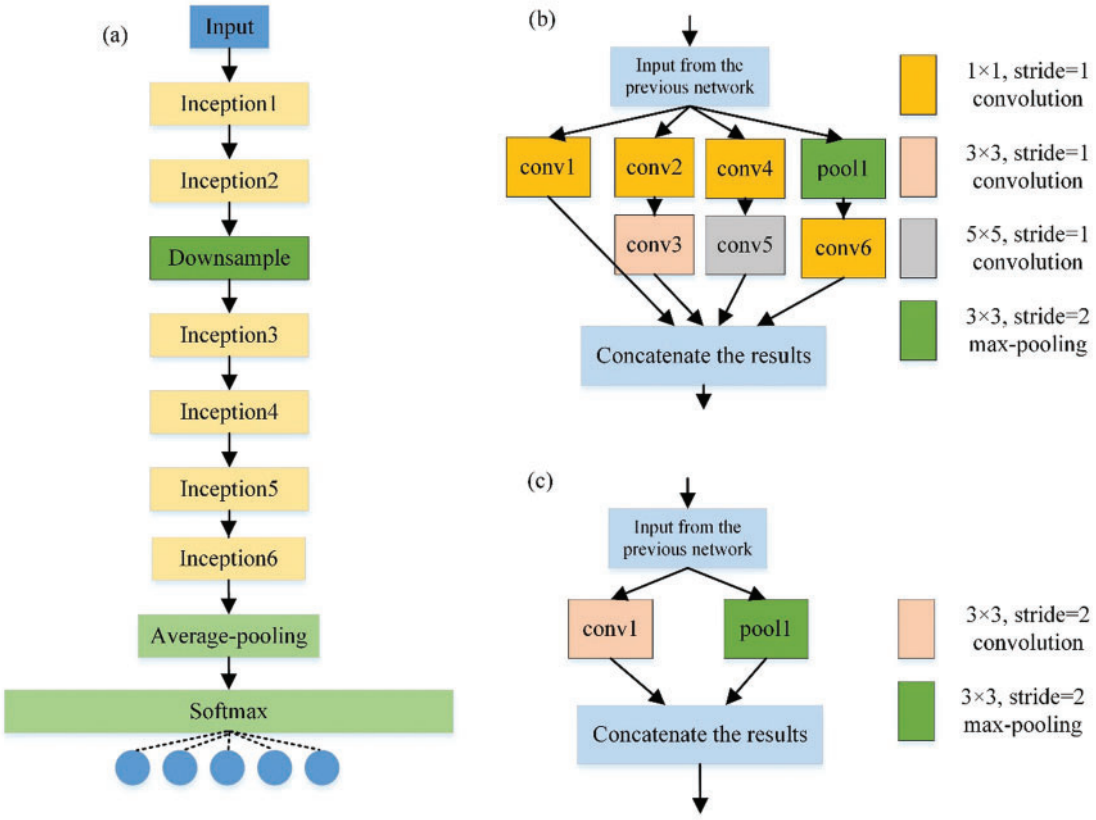

Figure 2: The general architecture of improved GoogLeNet model. (a) The architecture. (b) The inception module. (c) The downsample module

(2) FDL-IDM

FDL-IDM detects network attacks on empty packets without application-layer data. Among the common types of network attacks, SYN Flood and other unauthorized connections attack the transport-layer protocols. The data packets transmitted by these connections usually appear in the form of empty packets without application-layer data. The RDDL-IDM will remove empty packets when handling the packets. Therefore, the RDDL-IDM will not detect the attacks correctly, which can be addressed using the FDL-IDM.

FDL-IDM use CNN to address the detection problem perfectly. CNN is one of the most classical models in deep learning, where it is used to solve the problem of parameter explosion of high-dimensional data and improve the accuracy of classification. CNN network structure contains convolution-layer, pooling-layer and full connection-layer, as shown in Fig. 3.

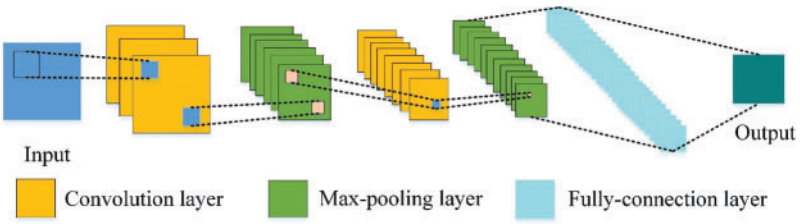

Figure 3: Typical structure of FDL-IDM

Generally, the convolution layer is used for feature extraction. The convolution method is used to extract features, where each neuron is only connected with a few other neurons in the upper layer; thus, the number of parameters must be reduced. In CNNs, the same set of connections share the same weight, which can effectively extract image features and reduce the amount of calculation resulted from the different weights.

Convolution is performed by formula (1) [56].

$$s = f(x \times w + b) \tag{1}$$

where s denotes the output data, known as feature mapping, x is the input sample data, w is the weight value of the kernel function, b is the bias value, and f is the activation function.

The convolution operation can efficiently extract the target image's features, but some redundant information will be in the generated feature map. The pooling-layer can remove the unobvious data in the features, enhance the density of effective data, reduce the number of parameters, and improve the model's robustness.

Using multiple convolution-layers and pooling-layers, the CNN model should use the fully connected layer to correlate the extracted features to be in a one-dimensional vector form. The vector is fed into a function named Softmax, which provided by the TensorFlow framework, to calculate the cross entropy for the final classification.

4 Experiments and Results

According to the datasets and methodology proposed, experiments are implemented, and then the results are analyzed.

4.1 Experimental Environment and Evaluation System

The software and hardware configurations of the experiments for the proposed method are given firstly, and then the evaluation system is explained.

(1) Experimental environment

The proposed method is executed using software and hardware, as described in Tab. 3.

(2) Evaluation system

In this experiment, accuracy (AC), false alarm (FA), and missing rate (MR) are used to evaluate the obtained results. These parameters are formulated as follows:

$$AC = \frac{TN + TP}{TN + TP + FN + FP} \times 100\% \tag{2}$$

$$FA = \frac{FP}{TN + FP} \times 100\% \tag{3}$$

$$MR = \frac{FN}{TP + FN} \times 100\% \tag{4}$$

where TN denotes the number of normal samples correctly classified, TP is the number of attack samples correctly classified, FN is the number of attack samples that are wrongly reported as normal, and FP is the number of normal samples that are wrongly reported as attacks.

Table 3: Software and hardware configurations

Operating system	64 bit windows 10 professional
Coding language	Python 3.7
Framework	TensorFlow-GPU v1.2
CPU	Intel(R) Core(TM) 2 Duo CPU T7300@2.00 GHz, 2.00 GHz
Memory	4.00 GB
GPU	NVIDIA GTX 1050Ti 4 GB
Hard disk	500 GB

4.2 Experiments

A series of experiments are carried out in accordance with the experimental environment and evaluation parameters described in Section 4.1. The NSL-KDD dataset is used to train FDL-IDM, and the UNSW-NB15 dataset is used to train RDDL-IDM.

(1) Training the FDL-IDM

The one-hot encoded NSL-KDD dataset on the basis of TensorFlow-GPU deep learning framework is used and trained using the CNNs structure, as shown in Fig. 3. The corresponding parameters are shown in Tab. 4. Tab. 5 shows the hyper parameters of FDL-IDM, such as the batch size, epochs, initial learning rate (ILR) and so on. The Adam optimizer was used in FDL-IDM. Subsequently, the parameters are saved in the corresponding network model file, named FDL-IDM.pb.

Table 4: Parameters when training FDL-IDM

Structure	Key function	Kernel size/Filters	Padding	Stride	Output size
Input	/	/	/	/	$11 \times 10 \times 1$
Conv1	conv2d	$3 \times 3 \times 1/32$	SAME	1	$11 \times 10 \times 32$
Pool1	max_pool	2×2	SAME	2	$6 \times 6 \times 32$
Conv2	conv2d	$2 \times 2 \times 32/64$	SAME	1	$6 \times 6 \times 64$
Pool2	max_pool	2×2	SAME	2	$3 \times 3 \times 64$
Output	softmax	/	/	/	2×1

(2) Training the RDDL-IDM

In this step, the UNSW-NB15 dataset is divided into training, validating, and testing datasets, within 8:1:1 proportion. The training and validating datasets are sent as input to the RDDL-IDM model. The hyper parameters of RDDL-IDM are shown in Tab. 4. Different with FDL-IDM, the batch size is set to 32, ILR is set to $1e-4$, and the value of decay is set to $1e-8$. Subsequently, the training parameters are saved in a file named RDDL-IDM.pb.

Table 5: Hyper parameters of FDL-IDM and RDDL-IDM

Hyper parameters	Batch size	Epochs	Initial learning rate	Decay	Optimizer
FDL-IDM	16	20	$1e-3$	$1e-4$	Adam
RDDL-IDM	32	20	$1e-4$	$1e-8$	Adam

(3) Deployment of the test environment

After trained the two models, the SDL-IDS is deployed on a Linux host, with the hardware parameters shown in Tab. 6, which is configured to enable the routing and forwarding functions. In addition, a virtual network experimental environment is created in the laboratory environment. The specific network topology is shown in Fig. 4.

Table 6: Hardware configurations of Linux host

CPU	Intel(R) Core(TM) i5-3470 CPU@3.20 GHz 3.60 Hz
Memory	8.00 GB
GPU	NVIDIA GTX 1050Ti 4 GB
Hard disk	2 TB

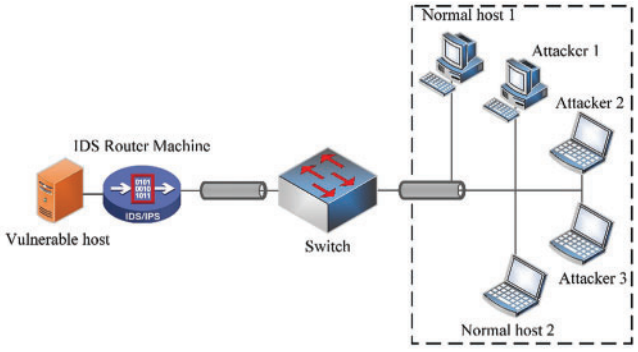

Figure 4: Network topological structure of experimental environment

In Fig. 4, the SDL-IDS installs a sniffer to monitor the network and selects different intrusion detection modules by detecting whether the monitored network packets contain application-layer data. If packets contain application-layer data, the data will be reduced and send to the

RDDL-IDM for detection. In addition, the corresponding classification results will be obtained. Otherwise, the feature extraction operation is performed with the utility named "kdd99_feature_-extractor" [57] according to the IP and Port information in the data packet. This utility can extract majority KDD'99 features from real-time traffic or .pcap file [57]. Subsequently, the current record is sent to the FDL-IDM for detection and obtain the corresponding classification results.

4.3 Results

In the virtual network attack environment, 982 attacks and 1502 normal visits are generated. The attack types are DoS, Exploits, Shellcode, and Generic. Tab. 7 shows the distribution of the attack types.

Table 7: Distribution of attack types

Attack type	DoS	Exploits	Shellcode	Generic
Count	127	341	319	195

After 20 epochs, the accuracy and loss curves of train and validation are show in Fig. 5, respectively. Same with FDL-IDM, the accuracy and loss of train and validation of RDDL-IDM are shown in Fig. 6.

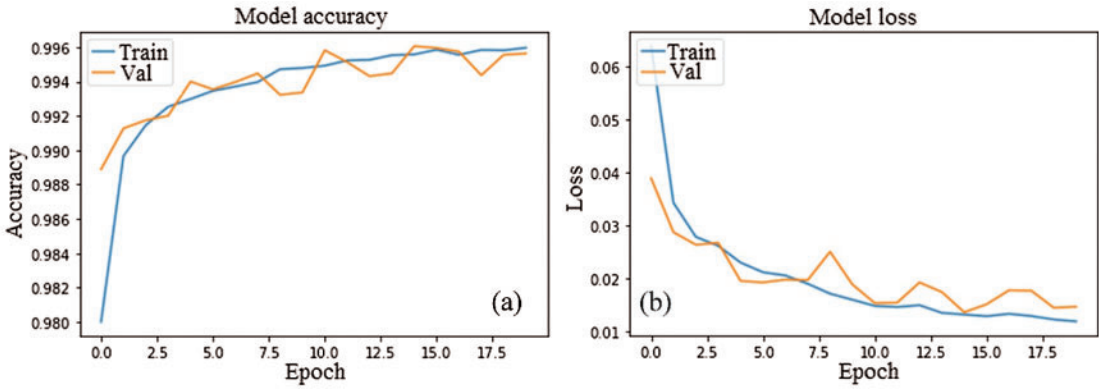

Figure 5: Accuracy and loss curves of FDL-IDM trained

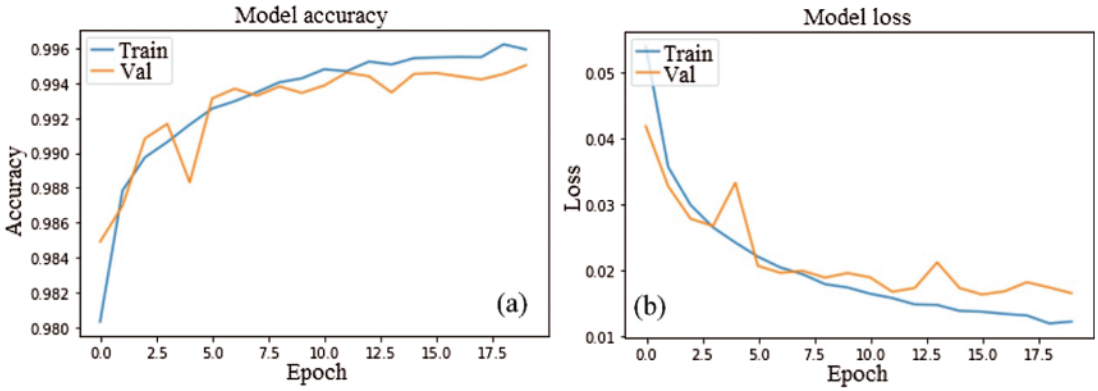

Figure 6: Accuracy and loss curves of RDDL-IDM trained

Tab. 8 shows the obtained results by the SDL-IDS. The results prove the robust performance of the SDL-IDS, where it improves the overall accuracy of intrusion detection to be 99.63%. The *MR* is reduced by up to 0.1% compared with RDDL-IDM and FDL-IDM. In addition, the SDL-IDS's *FA* is reduced by up to 0.2% compared with the single model.

Table 8: Experimental results

Model	RDDL-IDM/%	FDL-IDM/%	SDL-IDS/%
AC	99.4	99.5	99.63
FA	1.75	2.53	0.2
MR	0.23	0.58	0.1

Tab. 9 shows the corresponding confusion matrix. In the binary classification problem, the SDL-IDS improves the accuracy in detecting the unauthorized connections. In addition, the classification accuracy of the four types of attacks (i.e., DoS, Exploits, Shellcode, and Generic) is enhanced compared with the single model. The obtained overall classification accuracy of the four attack types reached up to 99.63%. However, some classifications are still missed, such as DoS.

Table 9: Confusion matrix of SDL-IDS

	Normal	DoS	Exploits	Shellcode	Generic
Normal	1502	0	0	0	0
DoS	1	120	3	2	1
Exploits	0	1	340	0	0
Shellcode	0	1	0	318	0
Generic	0	0	0	0	195

5 Conclusions

A step-based deep learning method for network intrusion detection is proposed. The proposed method combined the packets' raw data, used in RDDL-IDM, and the connections feature, used in FDL-IDM to achieve best overall five-class identification accuracy. The RDDL-IDM class can identify the packets with application-layer data, while the FDL-IDM class can identify the packets without application-layer data.

The proposed SDL-IDS is executed using an experimental environment. The proposed SDL-IDS enhanced the normal and unauthorized connections classification. In addition, the classification of unauthorized connections using the four attack types is improved and achieved results better than that of the single model. Generally, the proposed step-based deep learning intrusion detection system obtained the best results compared with the single model.

Possible future directions can be considered to enhance the proposed SDL-IDS by increasing the sample count and optimizing the super parameters. In addition, collect more DoS attack samples to improve the results.

Acknowledgement: The authors would like to express their gratitude to EditSprings (https://www.editsprings.com/) for the expert linguistic services provided.

Funding Statement: This work was supported by the Education Department of Jilin Province (No. JJKH20180518KJ), and Science and Technology Research Project of Jilin Business and Technology College (No. kz2018002).

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

1. Ayyagari, M. R., Kesswani, N., Kumar, M., Kumar, K. (2021). Intrusion detection techniques in network environment: A systematic review. *Wireless Networks, 27(2),* 1269–1285. DOI 10.1007/s11276-020-02529-3.
2. Vinayakumar, R., Alazab, M., Soman, K. P., Poornachandran, P., Al-Nemrat, A. et al. (2019). Deep learning approach for intelligent intrusion detection system. *IEEE Access, 7,* 41525–41550. DOI 10.1109/ACCESS.2019.2895334.
3. Tekerek, A. (2021). A novel architecture for web-based attack detection using convolutional neural network. *Computers & Security, 100(2),* 102096. DOI 10.1016/j.cose.2020.102096.
4. Kwon, D., Kim, H., Kim, J., Suh, S. C., Kim, I. et al. (2017). A survey of deep learning-based network anomaly detection. *Cluster Computing, 22(S1),* 949–961. DOI 10.1007/s10586-017-1117-8.
5. Adnan Khan, M., Rehman, A., Masood Khan, K., Ghamdi, A. Al, H.Almotiri, M. et al. (2020). Enhance intrusion detection in computer networks based on deep extreme learning machine. *Computers, Materials & Continua, 66(1),* 467–480. DOI 10.32604/cmc.2020.013121.
6. Devarajan, R., Rao, P. (2020). An efficient intrusion detection system by using behaviour profiling and statistical approach model. *The International Arab Journal of Information Technology, 18(1),* 114–124. DOI 10.34028/iajit.
7. Roesch, M. (1999). Snort-lightweight intrusion detection for networks. *Proceedings of the 13th Large Installation System Administration Conference*, pp. 229–238. Seattle, Washington, USA.
8. Hassan, M. M., Gumaei, A., Alsanad, A., Alrubaian, M., Fortino, G. (2020). A hybrid deep learning model for efficient intrusion detection in big data environment. *Information Sciences, 513(3),* 386–396. DOI 10.1016/j.ins.2019.10.069.
9. Assiri, A. (2020). Anomaly classification using genetic algorithm-based random forest model for network attack detection. *Computers, Materials & Continua, 66(1),* 767–778. DOI 10.32604/cmc.2020.013813.
10. Aljawarneh, S., Aldwairi, M., Yassein, M. B. (2018). Anomaly-based intrusion detection system through feature selection analysis and building hybrid efficient model. *Journal of Computational Science, 25(3),* 152–160. DOI 10.1016/j.jocs.2017.03.006.
11. Sivatha Sindhu, S. S., Geetha, S., Kannan, A. (2012). Decision tree based light weight intrusion detection using a wrapper approach. *Expert Systems with Applications, 39(1),* 129–141. DOI 10.1016/j.eswa.2011.06.013.
12. Hinton, G. E., Salakhutdinov, R. R. (2006). Reducing the dimensionality of data with neural networks. *Science, 313(5786),* 504–507. DOI 10.1126/science.1127647.
13. Krizhevsky, A., Sutskever, I., Hinton, G. (2012). ImageNet classification with deep convolutional neural networks. *Proceedings of the 25th International Conference on Neural Information Processing Systems*, vol. 1. Lake Tahoe, Nevada, USA: Curran Associates Inc.
14. Noda, K., Yamaguchi, Y., Nakadai, K., Okuno, H. G., Ogata, T. (2015). Audio-visual speech recognition using deep learning. *Applied Intelligence, 42(4),* 722–737. DOI 10.1007/s10489-014-0629-7.
15. Tara, N., Sainath, T. N., Kingsbury, B., Saon, G., Soltau, H. et al. (2015). Deep convolutional neural networks for large-scale speech tasks. *Neural Networks, 64,* 39–48.

16. Zhang, C. Y., Alexande, C. K. (2017). Machine learning interface for medical image analysis. *Journal of Digital Imaging, 30(5),* 615–621. DOI 10.1007/s10278-016-9910-0.
17. Ran, X., Xue, L., Zhang, Y., Liu, Z., Sang, X. et al. (2019). Rock classification from field image patches analyzed using a deep convolutional neural network. *Mathematics, 7(8),* 755. DOI 10.3390/math7080755.
18. Sermanet, P., Lecun, Y. (2011). Traffic sign recognition with multi-scale convolutional networks. *The 2011 International Joint Conference on Neural Networks*, pp. 2809–2813. San Jose, CA, USA.
19. Kuang, F., Xu, W., Zhang, S. (2014). A novel hybrid KPCA and SVM with GA model for intrusion detection. *Applied Soft Computing, 18(10),* 178–184. DOI 10.1016/j.asoc.2014.01.028.
20. Sheikhan, M., Jadidi, Z., Farrokhi, A. (2012). Intrusion detection using reduced-size RNN based on feature grouping. *Neural Computing & Applications, 21(6),* 1185–1190. DOI 10.1007/s00521-010-0487-0.
21. Xin, Y., Kong, L., Liu, Z., Chen, Y., Li, Y. et al. (2018). Machine learning and deep learning methods for cybersecurity. *IEEE Access, 6,* 35365–35381. DOI 10.1109/ACCESS.2018.2836950.
22. Vaiyapuri, T., Binbusayyis, A. (2020). Application of deep autoencoder as an one-class classifier for unsupervised network intrusion detection: A comparative evaluation. *PeerJ Computer Science, 6,* 327. DOI 10.7717/peerj-cs.327.
23. Ieracitano, C., Adeel, A., Morabito, F. C., Hussain, A. (2020). A novel statistical analysis and autoencoder driven intelligent intrusion detection approach. *Neurocomputing, 387(1),* 51–62. DOI 10.1016/j.neucom.2019.11.016.
24. Shone, N., Ngoc, T. N., Phai, V. D., Shi, Q. (2018). A deep learning approach to network intrusion detection. *IEEE Transactions on Emerging Topics in Computational Intelligence, 2(1),* 41–50. DOI 10.1109/TETCI.2017.2772792.
25. Yin, C., Zhu, Y., Fei, J., He, X. (2017). A deep learning approach for intrusion detection using recurrent neural networks. *IEEE Access, 5,* 21954–21961. DOI 10.1109/ACCESS.2017.2762418.
26. Li, X., Hu, Z., Xu, M., Wang, Y., Ma, J. (2021). Transfer learning based intrusion detection scheme for internet of vehicles. *Information Sciences, 547(14),* 119–135. DOI 10.1016/j.ins.2020.05.130.
27. Li, Z. P., Qin, Z., Huang, K., Yang, X., Ye, S. X. (2017). Intrusion detection using convolutional neural networks for representation learning. *International Conference on Neural Information Processing. Lecture Notes in Computer Science Lecture Notes in Computer Science*, vol. 10638, pp. 858–866. Guangzhou, China, Cham: Springer.
28. Churcher, A., Ullah, R., Ahmad, J., Ur Rehman, S., Masood, F. et al. (2021). An experimental analysis of attack classification using machine learning in IoT networks. *Sensors, 21(2),* 446. DOI 10.3390/s21020446.
29. Stiawan, D., Heryanto, A., Bardadi, A., Rini, D. P., Subroto, I. M. I. et al. (2021). An approach for optimizing ensemble intrusion detection systems. *IEEE Access, 9,* 6930–6947. DOI 10.1109/ACCESS.2020.3046246.
30. Venturi, A., Apruzzese, G., Andreolini, M., Colajanni, M., Marchetti, M. (2021). DReLAB-deep reinforcement learning adversarial botnet: A benchmark dataset for adversarial attacks against botnet intrusion detection systems. *Data in Brief, 34,* 106631. DOI 10.1016/j.dib.2020.106631.
31. Lv, L., Wang, W., Zhang, Z., Liu, X. (2020). A novel intrusion detection system based on an optimal hybrid kernel extreme learning machine. *Knowledge-Based Systems, 195,* 105648. DOI 10.1016/j.knosys.2020.105648.
32. Masdari, M., Khezri, H. (2020). A survey and taxonomy of the fuzzy signature-based intrusion detection systems. *Applied Soft Computing, 92,* 106301. DOI 10.1016/j.asoc.2020.106301.
33. Wu, W., Li, R., Xie, G., An, J., Bai, Y. et al. (2020). A survey of intrusion detection for in-vehicle networks. *IEEE Transactions on Intelligent Transportation Systems, 21(3),* 919–933. DOI 10.1109/TITS.2019.2908074.
34. Zhou, Y., Mazzuchi, T. A., Sarkani, S. (2020). M-AdaBoost-a based ensemble system for network intrusion detection. *Expert Systems with Applications, 162(6),* 113864. DOI 10.1016/j.eswa.2020.113864.
35. Adhi Tama, B., Nkenyereye, L., Lim, S. (2021). A stacking-based deep neural network approach for effective network anomaly detection. *Computers, Materials & Continua, 66(2),* 2217–2227. DOI 10.32604/cmc.2020.012432.
36. Yao, R., Wang, N., Liu, Z., Chen, P., Sheng, X. (2021). Intrusion detection system in the advanced metering infrastructure: A cross-layer feature-fusion CNN-LSTM-based approach. *Sensors (Basel), 21(2),* 626. DOI 10.3390/s21020626.

37. Zhang, K., Xian, M. (2018). Research on hybrid intrusion detection model based on DBN and TSVM. *Computer Applications and Software, 35,* 313–317+333. DOI 10.3969/j.issn.1000-386x.2018.05.056.
38. Wang, J., Tong, E., Niu, W., Liu, J., Zhao, D. (2018). Intrusion detection model based on CNN-NSVM. *Information and Communications Technologies, 6,* 48–55. https://oxxt.cbpt.cnki.net/WKE/WebPublication/paperDigest.aspx?paperID=f509ecdc-5843-4ab7-a7c2-43d589b7890b#.
39. Tavallaee, M., Bagheri, E., Lu, W., Ghorbani, A. (2009). A detailed analysis of the KDD CUP 99 data set. *Second IEEE Symposium on Computational Intelligence for Security and Defense Applications*, pp. 53–58. Ottawa, ON, Canada.
40. Stolfo, S., Fan, W., Lee, W., Prodromidis, A., Chan, P. (2020). Cost-based modeling for fraud and intrusion detection: Results from the JAM project. *DARPA Information Survivability Conference and Exposition*, Hilton Head, South Carolina, USA: IEEE.
41. Liang, J., Chen, J., Zhang, X., Zhou, Y., Lin, J. (2019). One-hot encoding and convolutional neural network based anomaly detection. *Journal of Tsinghua University (Science and Technology), 59,* 523–529. DOI 10.16511/j.cnki.qhdxxb.2018.25.061.
42. Ambusaidi, M. A., He, X., Nanda, P., Tan, Z. (2016). Building an intrusion detection system using a filter-based feature selection algorithm. *IEEE Transactions on Computers, 65(10),* 2986–2998. DOI 10.1109/TC.2016.2519914.
43. Wang, G. G., Tan, Y. (2019). Improving metaheuristic algorithms with information feedback models. *IEEE Transactions on Cybernetics, 49(2),* 542–555. DOI 10.1109/TCYB.2017.2780274.
44. Wang, G. G., Guo, L., Gandomi, A. H., Hao, G.-S., Wang, H. (2014). Chaotic krill herd algorithm. *Information Sciences, 274(2),* 17–34. DOI 10.1016/j.ins.2014.02.123.
45. Wang, G. G., Deb, S., Cui, Z. (2019). Monarch butterfly optimization. *Neural Computing and Applications, 31(7),* 1995–2014. DOI 10.1007/s00521-015-1923-y.
46. Gao, D., Wang, G. G., Pedrycz, W. (2020). Solving fuzzy job-shop scheduling problem using DE algorithm improved by a selection mechanism. *IEEE Transactions on Fuzzy Systems, 28(12),* 3265–3275. DOI 10.1109/TFUZZ.2020.3003506.
47. Chen, S. F., Chen, R., Wang, G. G., Gao, J., Arun, K. S. (2018). An adaptive large neighborhood search heuristic for dynamic vehicle routing problems. *Computers & Electrical Engineering, 67,* 596–607. DOI 10.1016/j.compeleceng.2018.02.049.
48. Sun, J., Miao, Z., Gong, D. W., Zeng, X. J., Li, J. Q. et al. (2020). Interval multi-objective optimization with memetic algorithms. *IEEE Transactions on Cybernetics, 50(8),* 3444–3457. DOI 10.1109/TCYB.2019.2908485.
49. Li, J., Lei, H., Amir, H. A., Wang, G. G. (2020). Elephant herding optimization: Variants, hybrids, and applications. *Mathematics, 8(9),* 1415. DOI 10.3390/math8091415.
50. Zhang, Y., Wang, G. G., Li, K. Q., Yeh, W. C., Jian, M. W. et al. (2020). Enhancing MOEA/D with information feedback models for large-scale many-objective optimization. *Information Sciences, 522(1),* 1–16. DOI 10.1016/j.ins.2020.02.066.
51. Louvieris, P., Clewley, N., Liu, X. (2013). Effects-based feature identification for network intrusion detection. *Neurocomputing, 121(3),* 265–273. DOI 10.1016/j.neucom.2013.04.038.
52. Eesa, A. S., Orman, Z., Brifcani, A. M. A. (2015). A novel feature-selection approach based on the cuttlefish optimization algorithm for intrusion detection systems. *Expert Systems with Applications, 42(5),* 2670–2679. DOI 10.1016/j.eswa.2014.11.009.
53. Moustafa, N., Slay, J. (2015). UNSW-NB15: A comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set). *Proceedings of Military Communications and Information Systems Conference*, pp. 1–6. Canberra, ACT, Australia: IEEE.
54. Moustafa, N., Slay, J. (2016). The evaluation of network anomaly detection systems: Statistical analysis of the UNSW-NB15 data set and the comparison with the KDD99 data set. *Information Security Journal: A Global Perspective, 25(1–3),* 18–31. DOI 10.1080/19393555.2015.1125974.
55. Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Rabinovich, A. (2015). Going deeper with convolutions. *2015 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–9. Boston, MA, USA.

56. Lecun, Y., Bengio, Y., Hinton, G. (2015). Deep learning. *Nature, 521(7553),* 436–444. DOI 10.1038/nature14539.
57. GitHub (2015). AI-IDS/kdd99_feature_extractor: Utility for extraction of subset of KDD '99 features from realtime network traffic or .pcap file. https://github.com/AI-IDS/kdd99_feature_extractor.