Copyright © 2019 Tech Science Press CMES, vol.120, no.1, pp.1-23, 2019

A Hierarchy Distributed-Agents Model for Network Risk Evaluation Based on Deep Learning

Jin Yang[1], Tao Li[1], Gang Liang[1, *], Wenbo He[2] and Yue Zhao[3]

Abstract: Deep Learning presents a critical capability to be geared into environments being constantly changed and ongoing learning dynamic, which is especially relevant in Network Intrusion Detection. In this paper, as enlightened by the theory of Deep Learning Neural Networks, Hierarchy Distributed-Agents Model for Network Risk Evaluation, a newly developed model, is proposed. The architecture taken on by the distributed-agents model are given, as well as the approach of analyzing network intrusion detection using Deep Learning, the mechanism of sharing hyper-parameters to improve the efficiency of learning is presented, and the hierarchical evaluative framework for Network Risk Evaluation of the proposed model is built. Furthermore, to examine the proposed model, a series of experiments were conducted in terms of NSL-KDD datasets. The proposed model was able to differentiate between normal and abnormal network activities with an accuracy of 97.60% on NSL-KDD datasets. As the results acquired from the experiment indicate, the model developed in this paper is characterized by high-speed and high-accuracy processing which shall offer a preferable solution with regard to the Risk Evaluation in Network.

Keywords: Network security, deep learning (DL), intrusion detection system (IDS), distributed agents.

1 Introduction

IDS refers to a system that automatically analyzes the network for malicious activity and policy violations, which is an abbreviated form of Intrusion Detection System. The firewall, evaluation, frangibility, virus detection, etc. are involved in this approach. The usual method for network intrusion detecting is Network Firewall. All these methods and sub-details under this approach are primarily dependent on the collection and analysis on the intrusion signatures or specimens taken on by viruses whereby the traditional technologies, inclusive of whitelists, characteristics analysis, blacklists, keyword filtering, statistical analysis, for example, Gupta et al. [Gupta and Raskar (2018)], etc. Yet, the mechanisms for defense being adopted currently have been unable to keep the infrastructures of network out of the cyber-threats arising from the rise of intensity and frequency in the threats [Raman, Somu, Kirthivasan et al. (2017)] to the network shall be

[1] College of Cyber Security, Department of Computing and Software, Sichuan University, Chengdu, China.

[2] McMaster University, Science and Technology on Communication Security Laboratory, Canada.

[3] Science and Technology on Communication Security Laboratory, Chengdu, China.

[*] Corresponding Author: Gang Liang. Email: jinnyang@163.com.

consequently lagged in the face of novel threats because of these approaches [Salo, Nassif and Essex (2019)]. The detection accuracy of the existing algorithm model is heavily dependent on the number of customized rules. But with the increase of regulations, the time to learn useful regulations increases exponentially. In theory, the more data the system trains, the more efficient and precise the rules that the system is learning become. Especially in the face of the current mass of data, adequate training is necessary. Unfortunately, in recent years, traditional algorithms can only process a small amount of data, let alone try to do high-speed processing and obtain adequate training for massive data. The inadequate training process leads to a decrease in the accuracy of the inspection. Given that the ability to learn independently and to be geared independently is lacked, merely the known network intrusions can be prevented by the traditional techniques, whereas the intrusions being unknown shall not be effectively reckoned with Selvakumar et al. [Selvakumar and Muneeswaran (2019)]. Consequently, the time to react and adapt shall be shrunk arising from these models in the face of the booming newly occurring network attacks.

Deep learning deemed to be a newly branched subject of machine learning, which is currently arousing more concerns and becomes another hotspot in the field of research for nature-inspired computational methodologies enlightened biologically. Deep learning shall primarily rely on its profound framework and ability for reckoning with considerable data. McCulloch et al. [McCulloch and Pitts (1943)] proposed Threshold Logic Theory in 1943, which paved the way for adopting neural networks in Artificial Intelligence. The Perceptron proposed by Rosenblatt [Rosenblatt (1958)] in 1958 used a linear threshold function, which is the first neural network. In 1965, Ivakhnenko et al. [Ivakhnenko and Lapa (1967)] created the first supervised feed-forward deep neural networks, then called Multilayer Perceptrons. In 1975, Werbos [Werbos (1975)] created a BP algorithm that effectively solved the XOR problem, which renewed interest in neural networks and learning. In 1986, to simulate neural processes, the connectionism model was initially described by Rumelhart et al. [Rumelhart and McClelland (1986)]. In the same year, Dechter [Dechter (1986)] first introduced the idea of Deep Learning to the machine learning community. In 1989, LeCun et al. [LeCun, Boser, Denker et al. (1989)] applied standard BP into automatic differentiation to identify the ZIP codes handwritten. In 2014, Schmidhuber [Schmidhuber (2014)] created multi-level hierarchies of recurrent neural networks to speed up supervised learning. In 1995, Hinton et al. [Hinton, Dayan, Frey et al. (1995)] and [Hinton (1994)] described the use of the wake-sleep algorithm and proved a network comprised of six layers overall connected would be likely to train. Furthermore, in 2006, Hinton et al. [Hinton, Osindero and Teh (2006)] and [Hinton (2006)] first proposed the full of Deep Learning conception and reported a significant breakthrough in feature extraction, which made Deep Learning generate considerable research interest in various fields [Bengio (2012)]. In 2012, Ciresan et al. [Cireşan, Meier and Schmidhuber (2012)] created Multi-column deep neural networks to recognize traffic signs and handwritten digits, a method which has already achieved human-competitiveness or even exceeded human performance. Currently, numerous and diverse researches and applications have already adopted Deep Learning. In 2011, Patidar et al. [Patidar and Sharma (2011)] created a Multilayer Perceptron model to detect fraudulent transactions through the neural network. In 2013, Vaswani et al. [Vaswani, Zhao, Fossum et al. (2013)] explored an application of Deep Neural Language Models to machine translation which can improve translation quality. In the same year, C. Z. Pan's paper has also adopted a Neural Network taking on multilayer characteristic for controlling an automatically driven surface vehicle [Pan, Lai, Yang et al. (2013)]. In 2014, Lee et al. [Lee and Choeh (2014)] developed a BP neural network model to evaluate how conducive and beneficial the reviews might be, to offer a tool seeking out the reviews of a given product taking on the benefits to the largest extent. In 2015, Ghiassi et al. [Ghiassi, Lio and Moon (2015)] developed a model using Multilayer Perceptrons of neural networks to forecast movie revenues during the pre-production stage. In the business domain, a deep neural model was established by De Oliveira et al. [De Oliveira, Nobre and Zarate (2013)] for the financial market for the prediction of how the stock price shall be varied in line with technical analysis in 2013. A model based on Deep Neural Networks was successfully developed by Iturriaga et al. [Iturriaga and Sanz (2015)] as to study how U.S. banks become insolvent in 2015, outstripping the performance of traditional models to forecast improverishment. In 2016, on the basis of a multi-layered deep learning network of an obstructed Boltzmann machine, an approach to detect fingerprint liveness was developed by Soowoong Kim et al. [Kim, Park, Song et al. (2016)]. And in the same year, on the basis of a deep learning network, a network traffic forecast approach was proposed by Nie et al. [Nie, Jiang, Guo et al. (2016)]. In 2017, Polson et al. [Polson and Sokolov (2017)] created a deep learning model to indicate how deep learning forecasts traffic flow in short-term accurately. Park et al. [Park, Kim, Kim et al. (2017)] proposed a deep learning-based player evaluation model by combining both quantitative game statistics and the qualitative analyses provided by news articles. The proposed system applied to a Korean professional baseball league (KBO) and it was shown to be capable of understanding the sentence polarity of news articles on player performances. Lee et al. [Lee, Chan, Mayo et al. (2017)] designed a hybrid feature extraction model for plant classification by using Convolutional Neural Networks (CNN) in 2017. Accordingly, plant classification systems shall be more precise in identification. The concept of deep learning was introduced into the classification of islanding and grid disturbance for the first time in Kong's paper in 2017 [Kong, Xu, Yan et al. (2017)]. He created a newly developed deep learning framework to detect and classify islanding or grid disturbance. The process of mass data has been effectuated and realized by virtue of the innovations of Deep Learning algorithms and GPU processing in recent years. Camps et al. [Camps, Sama, Martin et al. (2017)] first proposed a deep learning method for detecting FOG episodes in PD patients. In this paper, the deep learning model achieved 90% for the geometric mean between sensitivity and specificity, whereas the most updated approaches were unable to outstrip 83% for the same metric. Cybersecurity fields are concerned with Deep Learning as this approach has been successfully adopted in various fields of big data [LeCun, Bottou, Bengio et al. (1998); Hinton (2009); Goodfellow, Lee, Le et al. (2009); Salo, Nassif and Essex (2019); Mahbod, Schaefer, Ellinger et al. (2019)]. By virtue of its capability to extract high-level feature, as Deep Learning is adapted to detect the attack in cyberspace, this approach [Diro and Chilamkurti (2017)] shall be resilient to detect the small mutations or newly developed attacks.

In this paper, a newly developed approach is shed light on in this work to analyze the network for malicious activity and policy violations using Deep Learning. The proposed approach is motivated by applications of deep neural networks to IDS, more specifically for classifying regular network behavior or malicious activity. And on this basis, we create a network risk evaluation model. The contribution of the proposed research is threefold:

1. On the basis of Deep Neural Networks, a novel model to detect Intrusion is proposed in this work. The results attained in these experiments demonstrated convincingly that the effectiveness of Deep Learning in IDS. The model is proved to outperform the traditional machine learning method in the detection of abnormal network activities.
2. We combine a Distributed Agents method with deep learning method to reduce training time, and to improve accuracy and training efficiency. Previously, researchers rarely make such flexible designs in the field of Intrusion Detection using Deep Learning methods.
3. The selections and combinations of the Hyper Parameters have a great impact on the results of the detection accuracy. Through repeated experiments, we give better Weights of Hyper Parameters, which can provide a full play to the advantages of deep learning, and can further improve detection accuracy.

The proposed model was able to differentiate between normal and abnormal network activities with an accuracy of 97.60% on NSL-KDD datasets. The results attained from experiments indicate that the proposed model shall be more accurate than all of six learning methods in the classification of network intrusion behavior.

2 Related works

In 2016, an IDS was proposed by Kang et al. [Kang and Kang (2016)] in line with a DNN to ensure an in-vehicular network to be secured. This paper demonstrated that their proposed model is able to respond to the attack correspondingly and promptly, taking on an evidently optimized detection ratio. It is a newly developed approach, but its centralized processing method might limit its practicality in high-speed transmission, big data, and the heterogeneous Internet. In 2017, a distributed deep learning was proposed by Abebe in the light of a model to detect the attack on Internet of Things. Their experiments have shown that the model of deep learning shall more effectively detect the attack compared with its superficial counterparts. In Li et al. [Li, Ma and Jiao (2015)], based on the AutoEncoder and Deep Belief Networks, a scheme for detecting the hybrid malicious code was proposed by Li. As the results acquired from experiment indicate, their model in this paper outstrips single DBN in the accuracy for detection, as it reduced the time complexity and has better detection performance. In 2017, Bu et al. [Bu and Cho (2017)] utilized a Convolutional Neural-Learning Classifier System to detect intrusions on database, especially against insider attacks. As the results acquired from experiment bespoke, the proposed model outperformed other machine learning classifiers. In 2016, a deep learning method was adopted by Tang et al. [Tang, Mhamdi, McLernon et al. (2016)] to detect the anomaly on the basis of the flow under the condition of SDN. A model for Deep Neural Network was established, which took on six basic features for an IDS. The experiment confirmed that the deep learning approach was indicated to be remarkably potential as to be adopted to detect the anomaly in SDN environments on the basis of the flow.

Now in the IDS domain, on the basis of the deep learning neural networks, numerous research works have been conducted. Deep learning has several advantages that can be applied to IDS. Deep learning is able to derive features from raw signals automatically, in contrast with manually pre-designed statistical features [Dahiya and Srivastava (2018)]. This excellent feature allows the Deep Learning algorithm to recognize attacks more efficiently. Additionally, Deep Learning algorithm has better ability to deal with big data than traditional methods, especially in the face of network intrusion behavior analysis in a massive data network environment. Of course, Deep Learning algorithm has the disadvantages of a long-training time. When faced with the Intrusion Detection area which needs rapid response, the detection system will not resolve it in a timely manner. In addition, arising from the heterogeneity of the Internet, big data, unstructured data and other features, the traditional centralized response architecture cannot meet the requirements of current IDS. Moreover, the terrible network security situation requires that new IDS should be provided with higher detection accuracy and be more intelligent than traditional algorithms. These are problems that we want to solve in this paper. For this reason, a new method shall be presented to deal with detecting network attacks and network risk evaluation using deep learning.

3 Overview of deep learning

A Perceptron model can be used for two element classifications, but it cannot learn the complex nonlinear model. The neural network extends on the perceptron model and adds the hidden layers. In Deep Learning, there are multiple hidden layer stacks to enhance the expressive power of the model. So, Deep Learning is sometimes called a multilayer perceptron. The internal layer Deep Learning neural network includes 3 major categories, i.e., output layer, hidden layer, and input layer (Fig. 1).

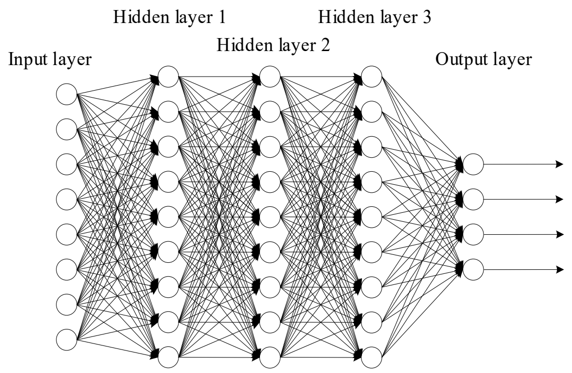


Figure 1: The structure of deep learning

Deep learning approach including the RNN, CNN and LSTM etc. can be used for the samples with different form, such as the time series, two-dimensional or three-dimensional objects. And back-propagation (BP) is one of the most important methods in Deep learning. BP algorithm has rekindled people's enthusiasm for neural network. It solves the problem of calculating the weights of hidden layer transfer in deep learning. BP counts as an approach for the calculation of the gradient taken on by the loss function in terms of the weights in an Artificial Neural Network-ANN [Rumerlhar (1986)]. BP serves as the typical approach for training ANNs to eliminate the objective function. A paper for analyzing the BP algorithm was jointly conducted by DE Rumelhart, GE Hinton and RJ Williams, accordingly boosting the research on neural network to a large extent [Ng, Ngiam, Foo et al. (2014)]. Hypothesize that a training set is established as $\{(x^{(1)}, y^{(1)}), \cdots, (x^{(m)}, y^{(m)})\}$ of *m* training examples. The proposed neural network can be trained whereby gradient descent in batch. The cost function regarding such single example shall be given as [Chellam, Ramanathan and Ramani (2018)]:

$$J(W, b; x, y) = \frac{1}{2}\left\|h_{w,b}(x) - y\right\|^2 \tag{1}$$

where a cost function is presented for squared and error. In line with a training set of *m* examples, the overall cost function shall be defined as:

$$J(W, b) = \left[\frac{1}{\mathrm{m}}\sum_{\mathrm{i}=1}^{m} J(W, b; x^{(i)}, y^{(i)})\right] + \frac{\lambda}{2}\sum_{\mathrm{l}=1}^{n_l-1}\sum_{\mathrm{i}=1}^{s_l}\sum_{\mathrm{j}=1}^{s_{l+1}}(w_{ji}^{(l)})^2$$

$$= \left[\frac{1}{\mathrm{m}}\sum_{\mathrm{i}=1}^{m}\left(\frac{1}{2}\left\|h_{w,b}(x^{(\mathrm{i})}) - y^{(\mathrm{i})}\right\|^2\right)\right] + \frac{\lambda}{2}\sum_{\mathrm{l}=1}^{n_l-1}\sum_{\mathrm{i}=1}^{s_l}\sum_{\mathrm{j}=1}^{s_{l+1}}(w_{ji}^{(l)})^2 \tag{2}$$

An average sum-of-squares error is regulated as the first term in the definition of $J(W, b)$. A regularization term is referred to as the second term which shall likely decline the magnitude taken on by the weights, and be conducive in the prevention of overfitting.

The function $J(W, b)$ shall be minimized. W, b shall be updated by one iteration of gradient descent as follows:

$$w_{i,j}^{(l)} = w_{i,j}^{(l)} - \alpha\frac{\partial J(w, b)}{\partial w_{i,j}^{(l)}} \tag{3}$$

$$b_i^{(l)} = b_i^{(l)} - \alpha\frac{\partial J(w, b)}{\partial b_i^{(l)}} \tag{4}$$

where α is the Learning Rate. In the step deemed as the crux, the foregoing partial derivatives shall be computed. In practice, these partial derivatives shall be effectively calculated by a BP algorithm. The BP is able to be adopted in the computation of $\frac{\partial J(w, b; x, y)}{\partial w_{i,j}^{(l)}}$ and $\frac{\partial J(w, b; x, y)}{\partial b_i^{(l)}}$. The derivative taken on by the general cost function $J(w, b)$ is as following:

$$\frac{\partial J(w, b)}{\partial w_{i,j}^{(l)}} = \left[\frac{1}{m}\sum_{i=1}^{m}\frac{\partial J(w, b; x^{(i)}, y^{(j)})}{\partial w_{i,j}^{(l)}}\right] + \lambda w_{i,j}^{(l)} \tag{5}$$

$$\frac{\partial J(w, b)}{\partial b_i^{(l)}} = \left[\frac{1}{m}\sum_{i=1}^{m}\frac{\partial J(w, b; x^{(i)}, y^{(j)})}{\partial b_i^{(l)}}\right] \tag{6}$$

4 Our model

Although the deep learning model has achieved excellent results in many research fields, there are few related models and algorithms in the field of network security, there are few relevant practical models and algorithms in the field of network security. Nowadays, with the development of technology of deep learning, the number of the instances which are applied deep learning to the field of manicous code detection has risen steadily. But there are few researches pay attention on the field of network risk evaluation.

What's more, arising from the variability and particularity of network intrusion behavior, a more appropriate model is needed. For the past ten years, some attacks turned out to be even more complicated, and a new approach has been upheld by IDS systems, and a significant segment of network defense has been replaced by IDS. The neural network system is complex and able to adapt, learn and organize independently, as well as carry out the parallel processing, memory, and anti-interference. Additionally, it takes on the capability to learn complex classification tasks. In essence, the analysis of network behavior characteristics is also a kind of classification problem. The Deep Learning model has many advantages and can handle considerable data. After training, the model has strong generalization abilities and can recognize unknown attacks and new attacks. These excellent features can be applied to the field of network security. So, we can elaborately construct a deep neural network system for intrusion detection, which can distinguish normal network behavior and abnormal behavior.

On the other hand, under the open and dynamic cloud environment, networks are becoming more and more divergent, dynamic, heterogeneous and so on. In order to adapt to the diverse application requirements and continuously changing network environments, the IDS now need to use flexible, distributed, dynamic, and adaptive Artificial Intelligence methods to effectively analyze the intrusion behaviors. The distributed information management is deemed as one among applications for agents attracting the most attention, especially in the fields of intelligent computation. Agents are able to mitigate the latency taken on by residual communication, prevent the intermediate data from network transmission, and accordingly more rapidly finish the entire task in contrast to a traditional solution regarding client/server. The most noteworthy characteristic taken on by agents refers to their transition from one host to another, comprehensive study of experience, and realization of the ongoing self-learn and ability for a corporation. On the basis of IDS concepts, promising solutions are presented by agents to establish systems for intelligent network security.

In view of the demands, this work proposed a self-organizing approach in the light of agents to realize the cooperative analysis of network intrusion behavior. So, a Hierarchy Distributed-Agents Model is required to be proposed to evaluate the risk faced or existing in Network whereby Deep Learning, viz. the HDAMNRE-DL in this chapter. In the proposed architecture, a structure with hierarchical property is proposed in this work for intelligent agents. To elevate the rate of detection and decrease the false positive ratio to detect the intrusion of the network, we adopt a multi-agent method to conduct information exchange between separateness agents of network intrusion perception.

The whole system consists of monitoring center and series of Agents from hosts and subnet. Agents in the subnet collect and report information about network threats in real time, and receive instructions from the monitoring center. The monitoring center carries on the big data analysis based on Deep Learning to all the collected network threat information, and work out of the whole network threat and its risk indicators. It can also determine the defense strategy of the whole network according to the risk indicators of the current network, and send it to each monitoring subnet to implement the defense of the whole network. And the process is cyclical. Each agent is independent of the detection unit, avoiding a correlation between each other and enhancing detection efficiency. The overview of the entire system architecture is shown in Fig. 2. The system functions and the agents' cooperation scheme are elucidated as follows. The model takes the autonomous agent as the organizational unit and is divided into 6 categories Agents: (1) Data Acquisition Agent. (2) Perception Agent. (3) Junction Agent. (4) Analyzer Agent: (5) Status Report Agent. (6) Management Agent.

(1) *Data Acquisition Agent*: Its function is to collect the data at high speed and preprocess the collected data. Data preprocessing seeks to increase the accuracy and efficiency for detecting, and to unify the detection results. Data acquisition agent has the function of data preprocessing, including sampling, smoothing aggregation, data generalization, standardization, data normalization operation, etc.

(2) *Perception Agent*: The model based on Deep Learning is used to analyze intrusion behaviors. CNN is mainly used for data analysis. These agents pertain to hosts and shall supervise numerous and diverse surrounding environments, primarily referring to the search of abnormal behavior. This is because such behavior is deemed as of the places where the information associated with intrusion is primarily located in. See section 3. Because of the heterogeneity of the network in the cloud environment, each Perception agent environment may be completely different, and the set of hyper-parameters corresponding to each agent may be also completely different.

(3) *Junction Agent*: The Junction *A*gent primarily carries out data exchange protection between Agents, including the exchange of hyper-parameters. It can speed up training and reduce training time. When new attack behaviors are discovered by a single Intrusion Perception Agent, the agent will broadcast among all network agents. To ensure the communication to be safe, the agent uses the encryption algorithm for communication, including key exchange and authentication process.

(4) *Analyzer Agent*: Analyzer Agent obtains the network attack information from each Perception Agent and evaluates the risk of the subnet or area. It includes the technology of subnet risk value and the calculation of the whole network risk value.

(5) *Status Report Agent*: A Status Report Agent is encompassed by numerous blocks, e.g., time-stamps, being consistent with the time for creating the alert messages, the time to detect intrusion, and alarm information, inclusive of memory status, swap status, processes status, user status, system status, CPU status, network flux, network connection and IP packets, etc. We can check the status index of a single host, agent and network segment in real time, and improve the status report function by the designated port. In order to ensure the normal operation of the whole system, if an agent fails, the Status Report Agents can inform the system administrator in time.

(6) *Management Agent*: The manager is a higher-level component of the architecture. In addition, it also includes the management of permission, role assignment, task

assignment, training schedule control and so on. The manager maintains the whole architectural structure and completes the information synthesis gathered by the Analyzer Agent and Status Report Agent.

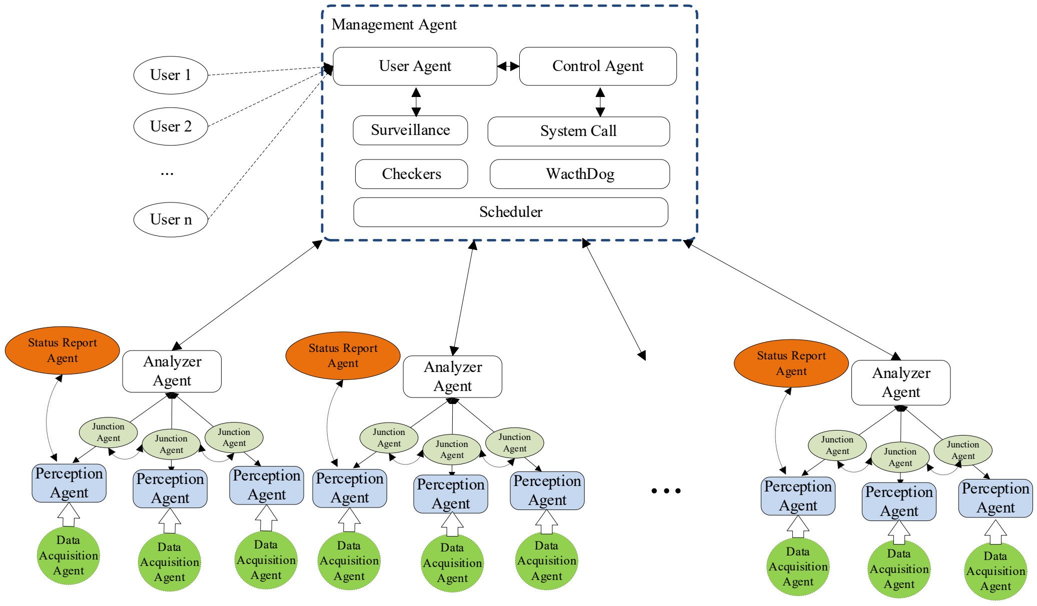


Figure 2: The functional architecture

Each Perception Agent node is a Deep Learning model, which contains 4 hidden layers Neurons as Fig. 3.

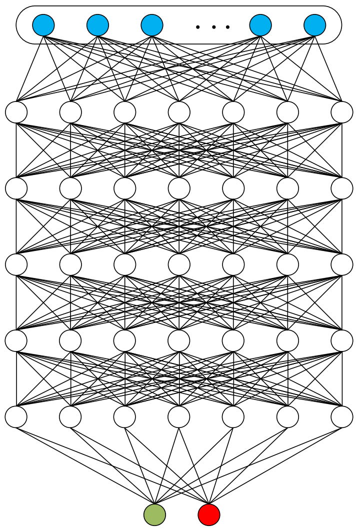


Figure 3: Each Perception Agent node contains 5 hidden layers Neurons

To train the proposed neural network, the parameter $W_{i,j}^{(1)}$ and $b_i^{(1)}$ shall be initialized to a small random value in the vicinity of zero. On that basis, an optimization algorithm shall be adopted, inclusive of batch gradient descent.

Then, each trained *Perception Agent* node reports the accuracy rate to the Superior level *Junction Agent*. If the accuracy of the *Junction Agent* is 10% higher than the accuracy rate of the original record, then the *Junction Agent* accepts hyper-parameters values that are submitted by the lower layer agent and broadcasts the super hyper-parameters values to each agent of the lower layer. In this way, the training of the agents is accelerated. To keep the entire system diverse, the agents satisfying the update condition have only 50% probability of updating the hyper-parameters.

After we get the network attack information from each Perception Agent, it is of great necessity for evaluating the risk faced by or existing in the network. The entire network risk counts as one among diversity (arising from numerous factors and stemmed from randomness). For this reason, the whole network risk evaluation involves diverse complex factors.

The values of the *Perception Agent* node indicate the intensity taken on by intrusion of the monitored-subnet or monitored-host. The more Agent nodes there are, the more comprehensive the evaluations of network risks are. As the proposed model is virtually associated with considerable factors relative to evaluation, the involved factors, i.e. importance of subnet, importance of area assets, importance of host assets, agent risk, subnet risk, area risk, and host risk, etc. are classified as to rationally and comprehensively ascertain the status of the network risk. The choice in the traditional evaluation system was commonly made in line with several factors, reckoning with other factors only for reference, and on that basis the possibility of the risk was described as low, medium or high. In this regard, numerous factors failing to be readily quantified were constantly ignored, which often led to the evaluation result lacking comprehensiveness and even becoming distorted.

To address this problem, the Analytic Hierarchy Process Principle that was proposed by Satty [Satty (1980)] is adopted in the proposed model to evaluate the network risk situation. Those complicated problems difficult to be reckoned with through adopting quantitative methods can be effectively solved whereby such approach. This approach takes on the characteristics as follows: first and foremost, capable of breaking complex questions down to gradations, and on that basis anatomizing gradually the layers more simply to indicate and reckon with the subjective judgments decision-makers whereby a quantitative format. Secondly, the weight taken on by the sequence indicating how the factors are relatively significant on each layer shall be calculated. By virtue of permutation all the layers, it shall calculate and sequence the relative weight taken on by all the factors. The quantitative and qualitative factors shall be synthetically considered in the course of the evaluation, and the AHP Principle shall be abided by, to offer rational approaches and instruments to comprehensively evaluate the status of network risk.

The following risk evaluation model is categorized into the overall object layer, the normal layer, and the factor layer. Additionally, to acquire the weights taken on by the foregoing factors, the AHP is adopted in this model. The crux idea refers to the comparison and estimation of the eigenvalue λ_i of the special equation of the matrix B through attaining a solution, On that basis, the peak eigenvalue $\lambda_{\max}$ shall be sought out, and its corresponding eigenvector $X = (x_1, x_2, \cdots, x_n)$ shall be attained. Eventually, the relative weight vector $W = (W_1, W_2, \cdots, W_n)$ shall be attained as all eigenvectors are incorporated as one. Briefly, the overall approach is able to be simplified as in the vicinity of four steps, viz. ① to build the structure model with hierarchy; ② to establish the matrices for evaluation; ③ to acquire relative weight taken on by factors in the case of a coincident standard; ④ to calculate the factors of incorporated weight base on each layer. In the following chapters, the evaluation shall be shed light on:

1) Establish Matrix A for identification: First and foremost, a matrix for identification must be established, as the relative significance of one group of elements is compared on the next layer with certain previous layer element constraints. In other words, the relative significance taken on by any pair of factors is accordingly indicated. In detail, a_{ij} denote the compared result of the *i*th factor and *j*th one, and A is the identify matrix.

$$A = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \cdots & \cdots & \cdots & \cdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{pmatrix} \tag{7}$$

where: $a_{ij} = 1$ if $i = j$ and $a_{ij} = 1/a_{ji}$ if $i \neq j$.

2) Calculating Weights: Secondly, the weight taken on by each factor shall be attained. As the matrix A for identification indicates, the maximum eigenvalue of the matrix $\lambda_{\max}$ can be attained. The maximum $\lambda_{\max}$ can be attained in line with the following conditions:

$$\begin{vmatrix} a_{11} - \lambda_1 & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} - \lambda_2 & \cdots & a_{2n} \\ \cdots & \cdots & \cdots & \cdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} - \lambda_n \end{vmatrix} = 0 \tag{8}$$

Acquire the corresponding eigenvector taken on by max eigenvalue of A, $X = (x_1, x_2, \cdots, x_n)$, let x_i be the weight of factor. On that basis, the unitary weights denoting W_i can be attained.

$$W = (W_1, W_2, \cdots, W_n)\left(x_1 / \sum_{i=1}^{n} x_i ,\, x_2 / \sum_{i=1}^{n} x_i ,\, \cdots ,\, x_n / \sum_{i=1}^{n} x_i \right) \tag{9}$$

3) Test of Consistency: Given that the evaluation is complex, and the individual knowledge is obstructed, the individual matrix for identification may fail to conform to the actual one may trigger error of subjective judgment. Yet, the consistency of the matrix A is required to be tested as following:

① Computing consistency value $C \cdot I$

$$C \cdot I = \frac{\lambda_{\max} - n}{n-1} \tag{10}$$

② Computing RI

The $\boldsymbol{R \cdot I}$ indicates mean consistency value able to be found in the reference and forms, such as: n=3, RI=0.58; n=4, RI=0.90; n=5, RI=1.12, … ; n=9, RI=1.45.

③ Computing consistency ratio $C \cdot R$

$$C \cdot R = \frac{C \cdot I}{R \cdot I} \tag{11}$$

As commonly considered, if $C \cdot R$ under 0.1, the consistency of the matrix A can be accepted, otherwise the matrix for identification *A must be modified*.

4) Calculating the Whole Weight Order: The whole weight order refers to the weight order in contrast to the elements in the existing layer and the highest layer. Each order of elements is regulated in rule layer, and the object layer and the values are regulated as $W_1, W_2, \cdots, W_n$, respectively. $W_1^j, W_2^j, \cdots, W_n^j$, are order designing layer to the rule layer and the values, then the whole order is

$$\vec{V} = W^j W = \begin{pmatrix} W_1^j & W_1^j & \cdots & W_1^j \\ W_2^j & W_2^j & \cdots & W_2^j \\ \cdots & \cdots & \cdots & \cdots \\ W_n^j & W_n^j & \cdots & W_n^j \end{pmatrix} \begin{pmatrix} W_1 \\ W_2 \\ \cdots \\ W_n \end{pmatrix} = \begin{pmatrix} V_1 \\ V_2 \\ \cdots \\ V_n \end{pmatrix} \tag{12}$$

Then, the risk level of the entire network shall indicate each host's value in the face of attacks adequately. Given that the host of each position turns out to be different, e.g., the hosts take on diverse properties as exerting influence on diverse social and even political values, operating another system for another user and offering different services, or exerting impacts on diverse economics.

The host or the subnet risk $r_m(t)$ can be defined:

$$r_m(t) = \frac{2}{1 + e^{-\sum \alpha_m \beta_n}} - 1 \tag{13}$$

where $\alpha_m(t)$ is the numbers of the mth host or subnet detecting attacks at time t. And, β_n $(0 \le \beta_n \le 1)$ is the risk coefficient of the nth kind of attacks in the network.

Let *Importance*$_{\mathrm{m}}$=$\sum_{k=1}^{K} (I_k \times V)$ be the importance coefficient of the mth host or subnet.

Then, the overall network risk level is attained: *R*(*t*)=∑(indicator value × indicator weight). In this regard, network risk *R*(*t*) status is attained, and network security is evaluated in real time.

$$R(t) = \tanh(\sum_{m=1}^{M}(Host_m^{'}\ or\ \ Subnet's\ \ risk \times \text{Importance}_m) \times Perception\ \ Agent\_Weight_m)$$

$$= \tanh(\sum_{m=1}^{M}(Host_m^{'}\ or\ \ Subnet's\ \ risk \times \sum_{k=1}^{K}(I_k \times \vec{V})) \times Perception\ \ Agent\_Weight_m)$$

$$= \tanh\left(\sum_{m=1}^{M}\left(r_m(t) \times \sum_{k=1}^{K}(I_k \times \vec{V})\right) \times Perception\ \ Agent\_Weight_m\right) \quad (14)$$

5 Experimental results

5.1 Dataset and experimental environment

In our study, we evaluate the model proposed in this paper on the well-known NSL-KDD [Chellam, Ramanathan and Ramani (2018)] which includes 125973 train and 22543 test records labeled anomaly and normal. Table 1 shows the detail of NSL-KDD dataset. This paper adopts the first 20% of the records in KDDTrain+ as the set of training, and adopts the KDDTest-21, KDDTest+ and KDDTest as sets of test. The specifications of the hosts adopted in the experiments are Core i7 6700k 6.4 GHz CPU, 64 GB RAM, 2 gtx1080 Ti GPUs. Ubuntu 16.04, Nvidia GTX 1080 Ti Device Driver*8, CUDA8.0, CUDNN 5.1, ANACONDA3, and Tensorflow 1.0 were adopted in each of the hosts.

Table 1: The distribution of the NSL-KDD dataset

	Normal	Anomaly	Total
Train	67343	58630	125973
Test	9711	12832	22543

5.2 Evaluation measures

In this paper, we use 2×2 Confusion Matrix [Stehman (1997)] which is a performance measurement for machine learning classification problem to evaluate our approach. As Tab. 2 shows, 2×2 confusion matrix is a table with 4 different combinations of predicted and actual values and extremely useful for measuring Recall, Precision.

Table 2: The general 2×2 confusion matrix

		Predicted	
		Positive	**Negative**
Actual	**Positive**	TP	FP
	Negative	FN	TN

Where TP is the case when the actual class of the data point was attack and the predicted is also attack, FP is the case when the actual class of the data point was attack and the predicted is normal, FN is the case when the actual class of the data point was normal and the predicted is attack, and TN is the case when the actual class of the data point was normal and the predicted is also normal.

Three standard metrics can be defined for the 2×2 matrix:

The Accuracy (AC) is the proportion of the total number of predictions that were correct. It is determined using the equation:

$$AC = \frac{TP + TN}{TP + FP + FN + TN} \tag{15}$$

The Recall or True Positive rate (TP) is the proportion of positive cases that were correctly identified, as calculated using the equation:

$$TP = \frac{TP}{TP + FN} \tag{16}$$

Finally, Precision (P) is the proportion of the predicted positive cases that were correct, as calculated using the equation:

$$P = \frac{TP}{TP + FP} \tag{17}$$

5.3 Performance of HDAMNRE-DL

The first set of tests was designed to ascertain the accuracy of the proposed model, and the final result showed the recognition accuracy is 97.6% (Fig. 4). The Hyper-parameter settings were as follows: hidden units=[20, 60, 30, 20], steps=20000, batch-size=5000, epoch=300, Learning Rate η=0.001.

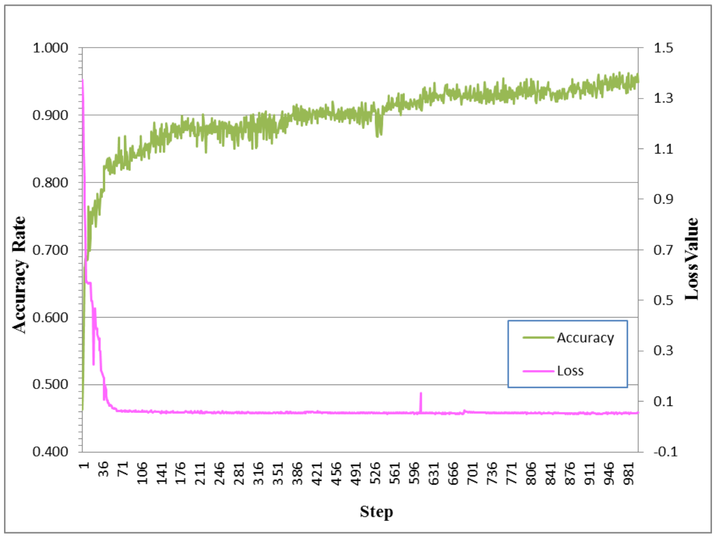


Figure 4: The accuracy and loss of our model

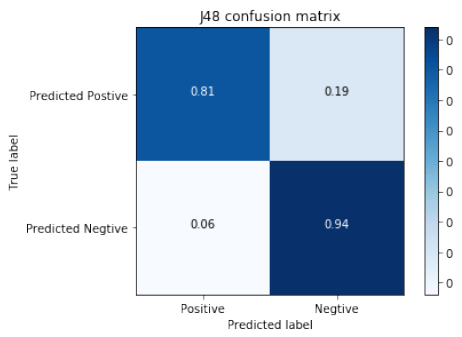
As indicated from Fig. 4, at the preliminary stage of the 50 steps, the recognition accuracy rapidly rose from 25% to about 65%, which pertained to the preliminary stage of learning, and the correct rate was not high at first. As this stage was merely initiated, the accuracy was very low, but was growing very fast. In the meantime, the loss value dropped rapidly from 1.0 to 0.5 at this stage. This is due to data exchange between many agents, which can speed up the learning of Hyper-parameters, and improve the speed of learning. From the 80 steps to the 100 steps, the learning rate went up steadily, and the correct rate was gradually raised from 65% to 95%. At the same time, the loss value decreased gradually from 0.5 to about 0.08, indicating that the training process gradually stabilized. From the 200 to the 800 steps, the accuracy rate increased from 95% to about 97.5%, the rate of correct rate of increase was slow, and the loss value remained at about 0.05, indicating that the study was basically completed. Eventually, we stopped training in the 1000 step.

5.4 Contrast experiment

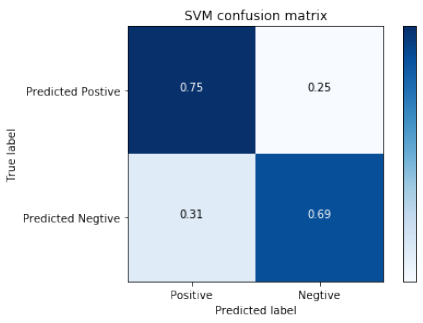
In order to evaluate the efficiency of our model proposed in this paper. We compare our approach with J48 decision tree, support vector machine, and Bayesian Network with correlation base feature selection, information Gain two feature selection strategies [Jamal (2017)]. Tab. 3 demonstrates the overall performance of the above-mentioned methods and our approach.

Table 3: Overall performance for the main methods and our approach

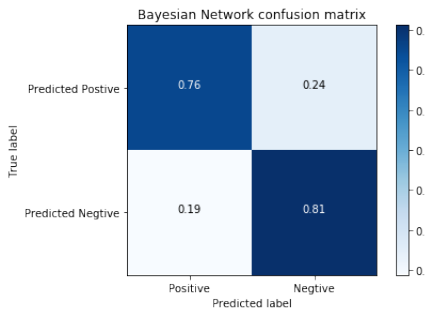
Classification method	Feature selection method/Numbers of hidden layer	Number of the select attribute	Detection accuracy
J48	Correlation	8	68.39%
		16	73.67%
		21	78.19%
	Info Gain	8	78.05%
		16	78.18%
		23	80.96%
SVM	Correlation	8	68.03%
		16	71.61%
		21	74.38%
	Info Gain	8	74.64%
		16	72.34%
		23	74.32%
Bayesian network	Correlation	8	65.58%
		16	70.49%
		21	75.61%
	Info Gain	8	71.05%
		16	72.63%
		23	72.10%
ANNs	2	41	86.00%
ANNs	3	41	87.50%
ANNs	4	41	89.00%
Our approach	4	41	97.60%



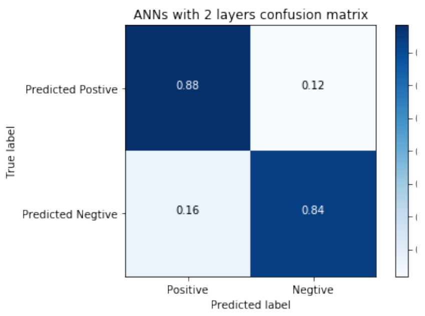
(A)



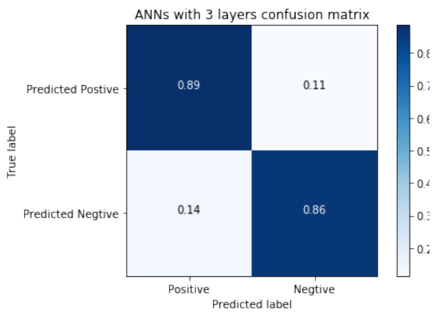
(B)



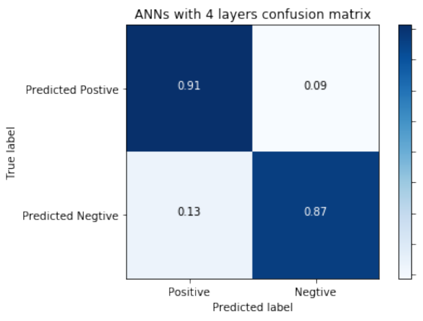
(C)



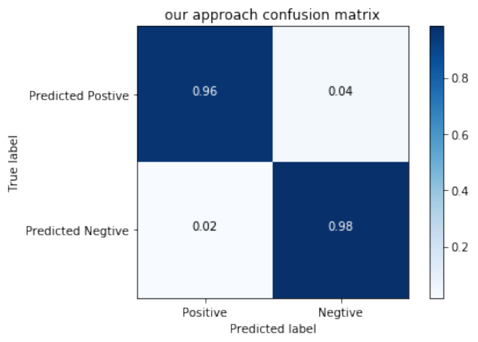
(D)



(E)



(F)



(G)

Figure 5: The confusion matrixes of different methods

Fig. 5 shows the confusion matrixes of the six main types methods (A-F) and our approach (G) based on NSL-KDD. And from the Tab. 2, the approach proposed in the paper shows a very promising performance and achieved 97.60% accuracy rate in the binary classification as compare to the best accuracies achieved by J48 (80.96%), SVM (74.64%), Bayesian network (75.61%), ANNs 2-layers (86.00%), ANNs 3-layers (89.00%) and ANNs 4-layers (89.00%), which proves the efficiency of our approach.

5.5 The influence of different parameters on the experimental results

(1) The affecting of learning-rateη on accuracy

The results attained when learning-rate η=0.0001, 0.001, 0.01 and 0.1 are presented in Fig. 6. As η increased, the average accuracy rate first increased and then decreased. When η=0.001, the final accuracy rate was the highest. In this regard, the choice of learning rate 0.001 is more rational.

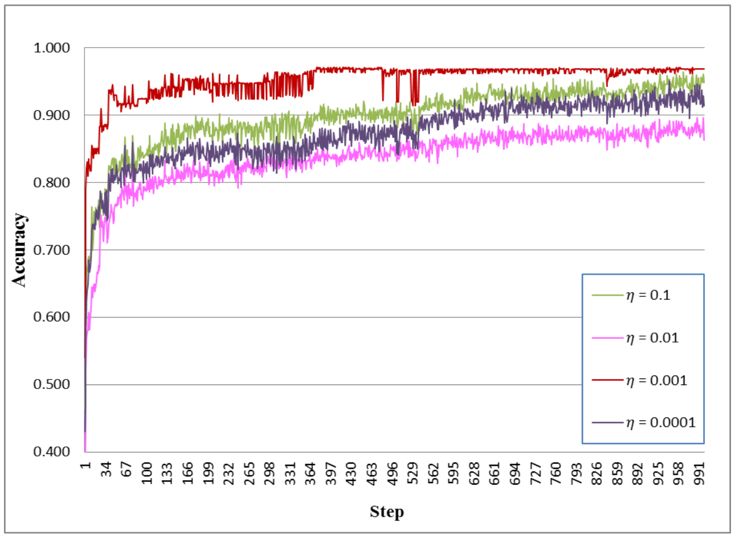


Figure 6: The affecting of learning-rate η on accuracy

(2) The affecting of hidden layer number on accuracy

To study the impact of the number of hidden layers, we varied the layer number from 3 to 7 and each layer had 100 neurons. As indicated in Fig. 7, the choice of 4 hidden layers is more rational. The results show that it did not improve the performance when the layer number was greater than 4.

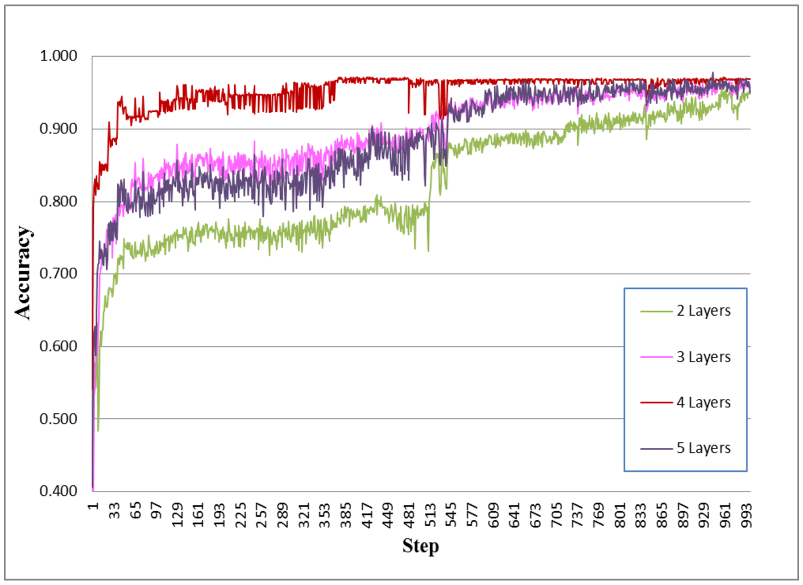


Figure 7: The affecting of hidden layer number on accuracy

(3) The affecting of batch size on accuracy

In this experiment, we tested the influence of batch size on the accuracy rate of proposed model. The results attained when batch-size=2000, 5000, 20000 and 50000 are presented in Fig. 8. As indicated in Fig. 8, the choice of batch-size=20000 is more rational. A too large batch-size shall make the training time become too long.

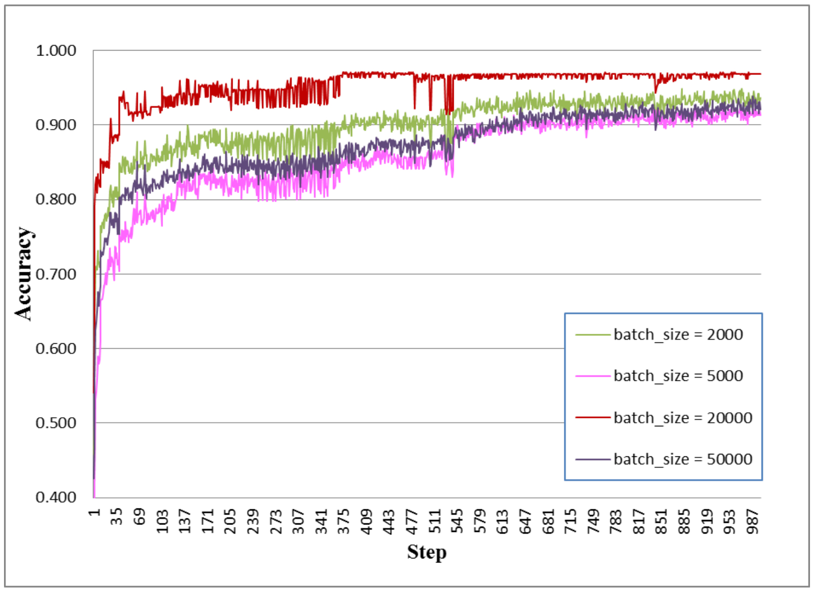


Figure 8: The affecting of batch size on accuracy

(4) The affecting of agent number on accuracy

The impact exerted by agent number on the accuracy rate of proposed model was ascertained in this experiment. The results attained when agent number=5, 10, 20 and 30 are presented in Fig. 9. The agent group which has high density was apparently able to attain well-trained information from a neighbor agent in Deep Learning to be able to adjust their initialization strategy quickly. As indicated in Fig. 8, the choice of agent number=20 is more rational. But when over 25 agent nodes, the rate of accuracy was decreased, and there was not much benefit to speeding up the training process. A too large agent number shall make the computation time become too long.

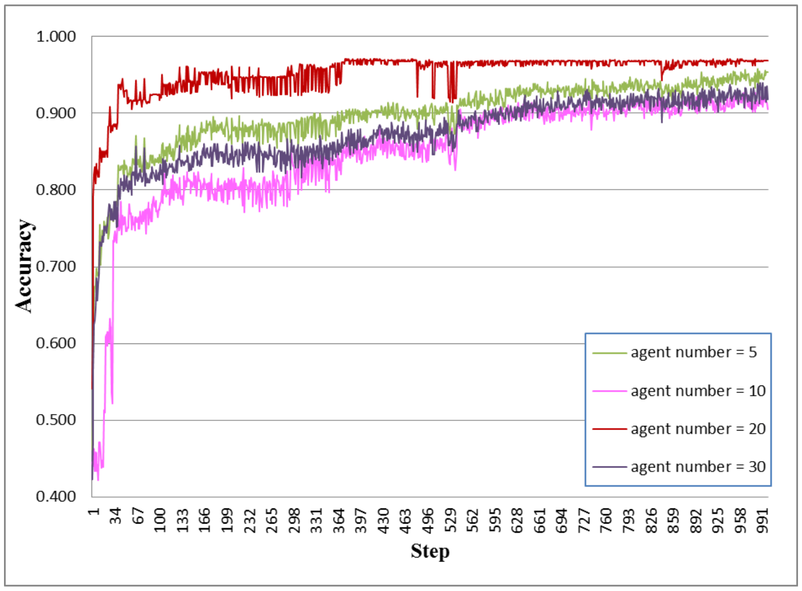


Figure 9: The affecting of agent number on accuracy

5.6 Results and analysis

The experiments have shown that our model based on deep learning is superior to traditional approaches. Here are reasons: First of all, the proposed model can train the data more thoroughly than traditional approaches. We can train all datasets instead of the 10% dataset. As the deep learning algorithm can handle a large amount of data well, it can ensure the data is fully trained. In addition, because of GPU use to accelerate the operation, it can significantly improve the training procedure. And using distributed intrusion behavior analysis can better improve detection accuracy rate. Since we put forward the Distributed-Agent, we can exchange the Hyper Parameters at the beginning of the training stage, which can greatly improve the speed and efficiency of training. In this regard, based on NSL-KDD datasets, testing had a good effect.

6 Conclusions

Traditional intrusion detection systems still have limitations on adaptability in the face of huge amounts of data and complex network environments. This paper presents a Network

Risk Evaluation model abiding by the Deep Learning theory. To increase the efficiency of packet for dumping, the distributed agents were adopted to real-time capture the traffic of the network. Additionally, the strengths taken on by this model over traditional models were shed light on. The mechanism of sharing hyper-parameters to improve the efficiency of learning was presented. Furthermore, a distributed framework taking on multiple hierarchies was adopted in the model to efficiently reckon with the network intrusion. Eventually, as the results attained from experiment indicate, the proposed model is characterized by self-adaptive abilities and real-time processing. Accordingly, a solution taking on enormous possibilities is offered to protect the security of the network.

Acknowledgement: This work is supported by the National Key Research and Development Program of China under Grant 2016YFB0800600, the Natural Science Foundation of China under Grant (No. 61872254 and No. U1736212), the Fundamental Research Funds for the central Universities (No. YJ201727, No. A0920502051815-98), and Academic and Technical Leaders' Training Support Fund of Sichuan Province (2016), and the research projects of the Humanity and Social Science Youth Foundation of Ministry of Education (13YJCZH021). We want to convey our grateful appreciation to the corresponding author of this paper, Gang Liang, who has offered advice with huge values in all stages when writing this essay to us.

References

Bengio, Y. (2012): Practical recommendations for gradient-based training of deep architectures. *Neural Networks: Tricks of the Trade*, pp. 437-478.

Bu, S. J.; Cho, S. B. (2017): A hybrid system of deep learning and learning classifier system for database intrusion detection. *International Conference on Hybrid Artificial Intelligence Systems*, pp. 615-625.

Camps, J.; Sama, A.; Martin, M.; Rodriguez-Martin, D.; Perez-Lopez, C. et al. (2017): Deep learning for freezing of gait detection in Parkinson's disease patients in their homes using a waist-worn inertial measurement unit. *Knowledge-Based Systems*, vol. 139, no. 1, pp. 119-131.

Chellam, A.; Ramanathan, L.; Ramani, S. (2018): Intrusion detection in computer networks using lazy learning algorithm. *Procedia Computer Science*, vol. 132, pp. 928-936.

Cireşan, D.; Meier, U.; Schmidhuber, J. (2012): Multi-column deep neural networks for image classification. *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3642-3649.

Dahiya, P.; Srivastava, D. K. (2018): Network intrusion detection in big dataset using spark. *Procedia Computer Science*, vol. 132, pp. 253-262.

De Oliveira, F. A.; Nobre, C. N.; Zarate, L. E. (2013): Applying artificial neural networks to prediction of stock price and improvement of the directional prediction index-case study of petr4, Petrobras, Brazil. *Expert Systems with Applications*, vol. 40, no. 18, pp. 7596-7606.

Dechter, R. (1986): Learning while searching in constraint-satisfaction problems. *National Conference on Artificial Intelligence Philadelphia*, vol. 119, no. 98, pp. 178-185.

Diro, A. A.; Chilamkurti, N. (2017): Distributed attack detection scheme using deep learning approach for internet of things. *Future Generation Computer Systems*, vol. 82, pp. 761-768.

Ghiassi, M.; Lio, D.; Moon, B. (2015): Preproduction forecasting of movie revenues with a dynamic artificial neural network. *Expert Systems with Applications*, vol. 42, no. 6, pp. 3176-3193

Goodfellow, I.; Lee, H.; Le, Q. V.; Saxe, A.; Ng, A. Y. (2009): Measuring invariances in deep networks. *Advances in Neural Information Processing Systems*, pp. 646-654.

Gupta, O.; Raskar, R. (2018): Distributed learning of deep neural network over multiple agents. *Journal of Network & Computer Applications*, vol. 1, pp. 1-8.

Hinton, G. E. (1994): Autoencoders, minimum description length, and helmholtz free energy. *Advances in Neural Information Processing Systems*, vol. 6, pp. 3.

Hinton, G. E. (2006): Reducing the dimensionality of data with neural networks. *Science*, vol. 313, no. 5786, pp. 504-507.

Hinton, G. E. (2009): Deep belief networks. *Scholarpedia*, vol. 4, no. 5, pp. 5947.

Hinton, G. E.; Dayan, P.; Frey, B. J.; Neal, R. M. (1995): The wake-sleep algorithm for unsupervised neural networks. *Science*, vol. 268, no. 5214, pp. 1158-1161.

Hinton, G. E.; Osindero, S.; Teh, Y. W. (2006): A fast learning algorithm for deep belief nets. *Neural Computation*, vol. 18, no. 7, pp. 1527-1554.

Iturriaga, F. J. L.; Sanz, I. P. (2015): Bankruptcy visualization and prediction using neural networks: a study of US commercial banks. *Expert Systems with Applications*, vol. 42, no. 6, pp. 2857-2869.

Ivakhnenko, A. G.; Lapa, V. G. (1967): Cybernetics and forecasting techniques. *American Elsevier Publishing Co. Bibliography*, pp. 163-166.

Jamal, H. (2017). NSL-KDD dataset classification using five classification methods and three feature selection strategies. *Journal of Advanced Computer Science and Technology Research*, vol. 7, no. 1, pp. 15-28.

Kang, M. J.; Kang, J. W. (2016): Intrusion detection system using deep neural network for in-vehicle network security. *PloS One*, vol. 11, no. 6.

Kim, S.; Park, B.; Song, B. S.; Yang, S. (2016): Deep belief network based statistical feature learning for fingerprint liveness detection. *Pattern Recognition Letters*, vol. 77, pp. 58-65.

Kong, X.; Xu, X.; Yan, Z.; Chen, S.; Yang, H. et al. (2017): Deep learning hybrid method for islanding detection in distributed generation. *Applied Energy*, pp. 19-31.

LeCun, Y.; Boser, B.; Denker, J. S.; Henderson, D.; Howard, R. E. et al. (1989): Backpropagation applied to handwritten zip code recognition. *Neural Computation*, vol. 1, no. 4, pp. 541-551.

LeCun, Y.; Bottou, L.; Bengio, Y.; Haffner, P. (1998): Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278-2324.

Lee, S. H.; Chan, C. S.; Mayo, S. J.; Remagnino, P. (2017): How deep learning extracts and learns leaf features for plant classification. *Pattern Recognition*, vol. 71, pp. 1-13.

Lee, S.; Choeh, J. Y. (2014): Predicting the helpfulness of online reviews using multilayer perceptron neural networks. *Expert Systems with Applications*, vol. 41, no. 6, pp. 3041-3046.

Li, Y.; Ma, R.; Jiao, R. (2015): A hybrid malicious code detection method based on deep learning. *International Journal of Information Security*, vol. 9, pp. 205-216.

Mahbod, A.; Schaefer, G.; Ellinger, I.; Ecker, R.; Pitiot, A. et al. (2019): Fusing fine-tuned deep features for skin lesion classification. *Computerized Medical Imaging and Graphics*, vol. 71, pp. 19-29.

McCulloch, W. S.; Pitts, W. (1943): A logical calculus of ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics*, vol. 5, no. 4, pp. 115-133.

Ng, A.; Ngiam, J.; Foo, C. Y.; Mai, Y.; Suen, C. (2014): UFLDL Tutorial. http://ufldl.stanford.edu/wiki/index.php/UFLDL_Tutorial.

Nie, L.; Jiang, D.; Guo, L.; Yu, S. (2016): Traffic matrix prediction and estimation based on deep learning in largescale IP backbone networks. *Journal of Network and Computer Applications*, vol. 76, pp. 16-22.

Pan, C. Z.; Lai, X. Z.; Yang, S. X.; Wu, M. (2013): An efficient neural network approach to tracking control of an autonomous surface vehicle with unknown dynamics. *Expert Systems with Applications*, vol. 40, no. 5, 1629-1635.

Park, Y. J.; Kim, H. S.; Kim, D.; Lee, H.; Kim, S. B. et al. (2017): A deep learning-based sports player evaluation model based on game statistics and news articles. *Knowledge-Based Systems*, vol. 138, no. 15, pp. 15-26.

Patidar, R.; Sharma, L. (2011): Credit card fraud detection using neural network. *International Journal of Soft Computing and Engineering*, vol. 1, pp. 32-38.

Polson, N. G.; Sokolov, V. O. (2017): Deep learning for short-term traffic flow prediction. *Transportation Research*, vol. 79, pp. 1-17.

Raman, M. G.; Somu, N.; Kirthivasan, K.; Liscano, R.; Sriram, V. S. (2017): An efficient intrusion detection system based on hypergraph-Genetic algorithm for parameter, Optimization and feature selection in support vector machine. *Knowledge-Based Systems*, vol. 134, no. 15, pp. 1-12.

Rosenblatt, F. (1958): The Perceptron: a probabilistic model for information storage and organization in the brain. *Psychological Review*, vol. 65, no. 6, pp. 386-408.

Rumelhart, D. E.; McClelland, J. L. (1986): Parallel distributed processing. *Explorations in the Microstructure of Cognition.* Cambridge MIT Press.

Rumerlhar, D. E. (1986): Learning representations by back-propagating errors. *Nature (London)*, vol. 323, pp. 533-536.

Salo, F.; Nassif, A. B.; Essex, A. (2019): Dimensionality reduction with IG-PCA and ensemble classifier for network intrusion detection. *Computer Networks*, vol. 148, pp. 164-175.

Satty, T. (1980): *The Analytic Hierarchy Process*. McGraw-Hill, New York.

Schmidhuber, J. (2014): Learning complex, extended sequences using the principle of history compression. *Neural Computation*, vol. 4, no. 2, pp. 234-242.

Selvakumar, B.; Muneeswaran, K. (2019): Firefly algorithm based feature selection for network intrusion detection. *Computers Security*, vol. 81, pp. 148-155.

Stehman, S. V. (1997): Selecting and interpreting measures of thematic classification accuracy. *Remote Sensing of Environment*, vol. 62, no. 1, pp. 77-89.

Tang, T. A.; Mhamdi, L.; McLernon, D.; Zaidi, S. A. R.; Ghogho, M. (2016): Deep learning approach for network intrusion detection in software defined networking. *International Conference on Wireless Networks & Mobile Communications*, pp. 2-17.

UNB. NSL-KDD data. http://nsl.cs.unb.ca/NSL-KDD/.

Vaswani, A.; Zhao, Y.; Fossum, V.; Chiang, D. (2013): Decoding with large-scale neural language models improves translation. *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pp. 18-21.

Werbos, P. J. (1975): Beyond regression: new tools for prediction and analysis in the behavioral. *Sciences*, vol. 29, no. 18, pp. 65-78.