Intellectual model for classification of network cybersecurity events

Tetiana Babenko, Grygorii Hnatiienko, Vira Vialkova, Andrii Bigdan

Taras Shevchenko National University, Volodymyrska St. 60, Kyiv, 01103, Ukraine

Abstract

Security experts are required to detect and determine appropriate countermeasures against modern computer attacks. Despite the increasing detection of exploits and vulnerabilities, methods of defense remain notably slower. However, this is still an open research issue. Our paper discusses our research in a specific field of network attack identification, leveraging neural networks, in particular a multilayer perceptron, to identify and predict future network security events based on previous observations. To ensure the quality of the learning process and to obtain the desired generalization of the model, 4 million records accumulated within 7 days by the Canadian Institute of Cybersecurity were used. Our result suggests that neural network models based on multilayer perceptron can be used, after refinement, to identify and predict network security events.

Keywords

Cybersecurity, Security System, Neural Network, Prediction, Network Security

1. Introduction

Each year, 111 trillion lines are added to the total number of software code, with each line potentially likely to be a new vulnerability and, therefore, a zero-day attack can be implemented. At the same time, technologies used by attackers to attack computer systems and networks become more and more complex and advanced [1]. Several ways for solving this issue are being explored, including technologies that allow the creation of secure software [2]. For this purpose, for the automation of processes of controlling security events in info systems, traditionally Intrusion Detection or Protection System (IDS / IPS) are used, the main task of which is to automate the process of identifying attacks [3-5] or improper uses [6; 7]. These systems, depending on the technology used to detect attacks, are usually divided into two main groups: malicious user behavior detection systems; and systems for detecting abnormal behavior of a computer system. In the first case, comparing the attack pattern with the flow of events, the second one compares the pattern of the normal behavior of the system with the flow of events. In this case, it is generally believed that the task of detecting intrusions in TCP/IP networks is reduced to recognition tasks [8-10]:

- structural signs (signatures) of known types of attacks;
- invariant signs of the structure of correct computational processes;
- correlation signs of the normal functioning of distributed computing systems. In the case of an issue with the recognition of network anomalies, there are some complexities that are mainly related to the increasing demand for identifying previously unknown attacks and destructive influences, which in its turn requires:

1. construction of reference sets of a normal (semantically correct) profile of the system's behavior in conditions of uncertainty of environmental influences;

CMiGIN 2022: 2nd International Conference on Conflict Management in Global Information Networks, November 30, 2022, Kyiv, Ukraine
EMAIL: babenkot@ua.fm (T. Babenko); g.gna5@ukr.net (G. Hnatiienko); veravialkova@mail.com (V. Vialkova); abigdan@gmail.com (A.Bigdan)
ORCID: 0000-0003-1184-9483 (T. Babenko); 0000-0002-0465-5018 (G. Hnatiienko); 0000-0001-9109-0280 (V. Vialkova); 0000-0002-2940-6085 (A. Bigdan)

2. definition of necessary and sufficient informative features;
3. construction of rules for determining anomalies.

As a rule, the detection of network anomalies is carried out according to the scheme in which the following functional blocks are [11]:

- analysis of the information contained in the headings of IP datagrams;
- construction of the prognosticating;
- search and estimation of anomalies;
- response to anomaly;
- filling and / or editing the IDS / IPS rules base.

The collected statistical information is used for the construction of a mathematical model for forecasting traffic based on cyclic analysis of time series. This model allows you to predict the network load based on the frequency search in network traffic.

In all cases, IDS might not be able to detect an intrusion, since they will not be able to distinguish it from the background white noise, which exists in any system due to weaknesses in the packet analysis tool or lack of signature of the corresponding attack, etc.

Consequently, most of the typical ways of identifying attacks and countering them have low accuracy and speed and do not allow effective counteracting of both known attacks and zero-day attacks. Therefore, many different technologies are being developed to protect computer systems [12] and networks based on data mining technologies and neural networks [13-17]. This is due to the neural network structure's ability to solve tasks that are difficult to formalize, as well as its ability to learn, self-organize, and generalize. This approach allows us to obtain models that can quickly adapt to the environment and allow us to predict the development of the process based on the property of generalization [18; 19].

Artificial intelligence is a broad term based on the imitation of human abilities by computers: to feel, understand, and react.

In the field of computer science known as machine learning, statistical techniques are frequently used to help computers "learn" (i.e., gradually improve their performance at a given job) [20].

Data Sciences – to apply (use) machine learning algorithms, it is necessary to define data sets, choose pertinent variables and metrics, and carry out a variety of information engineering tasks, such as looking for hidden dependencies, gathering data, training it, integrating it, visualizing it, and evaluating algorithm performance, among others.

Each learning model should be based on a certain algorithm. These include classification, clustering, associative rules, in-depth learning, regression, and pattern matching. The choice of algorithm depends on the ultimate goal, which is set as a goal to achieve. The model of cybersecurity event identification, considered in this research, is based on supervised training and on classification algorithms of neural networks.

2. Material and methods

While solving the task of classifying events that occur in the process of network interaction, some issues arise, mostly associated with a need to account for unknown attacks and destructive influences, which in turn needs: building benchmark sets of normal (semantically correct) profile behaviors of the system in conditions of uncertainty of environmental influences, defining the necessary and sufficient informative signs and building rules for identifying anomalies [21; 22].

A similar and less complex task was performed for SQL (Structured Query Language) injection attacks [23]. An attack known as SQL injection involves changing database queries by taking advantage of weaknesses in online applications. A successful attack enables the attacker to collect, alter, or even delete sensitive information.

To synthesize and analyze the SQL injection identification model, we prepared preliminary training, control, and test datasets. The training dataset contained the parameters of the training object, and the parameters were selected heuristically based on the analysis of significant attack features that may contain a URL.

Artificial neural networks are mathematical models and their software or hardware implementations. The term was coined in the study of processes in the brain and attempts to model these processes.

Interpreting sensory data using a kind of machine perception by labeling or grouping input data is the basic principle of building neural networks. Any data is translated into vectors containing recognized patterns, which are numerical [24].

In a neural network, each node has inputs and an output. A neuron has two modes: training and use. During training, the neuron learns to respond to specific input patterns. In use mode, the neuron responds to input and generates an output. If the input is unusual, the neuron uses its activation function to decide whether to activate itself or not.

The relevant weight of each input signal is calculated based on the input data. If this value exceeds the threshold, the neuron is triggered.

Neural models work exclusively with numerical data presented in a certain numerical range, so at the first stage of the study, a numerical URL classifier was developed. This subroutine converts the URL into a binary value and sets 1 (the logical value "true") if the parameter in the URL is related to the attack, and 0 (the logical value "false") otherwise. Thus, an input vector was generated for each URL, and the output vector can be represented as:

$$X^T = [x_1 x_2 \cdots x_n],$$

where *n* is the number of query parameters used in the SQL pattern of the URL.

Thus, the neural network model will have n neurons as input. Each vector is defined by a value: Benign (0) means no attack, while Injection (1) means an attack. To divide the input vectors into two classes with values of 0 and 1, it is enough to have only one neuron at the output.

As a result, we obtained a relative error of the simulated model of no more than 5% for both the control and test samples. So, we may apply such an approach to a wider range of security events and attacks.

The purpose of this research is to study opportunities for using artificial neural networks, in particular, the perceptron of Rumelhart (a separate case of the perceptron of Rosenblatt) for the identification of network attacks and predicting events connected with network security [25; 26]. To ensure the quality of the training process and to obtain the desired generalization of the model's properties, it is necessary to have a significant number of implementation examples of the relevant attacks. For experimental purposes, 4 million records were collected that were accumulated within 7 days by the Canadian Institute of Cybersecurity. The networking infrastructure of the attacker consisted of 50 machines. The victim organization consisted of 5 departments, and each of them used 420 user nodes and 30 servers. The dataset contains information about the intercepted network traffic and system logs of each machine of the victim organization. The data processing scheme is presented in Figure 1.

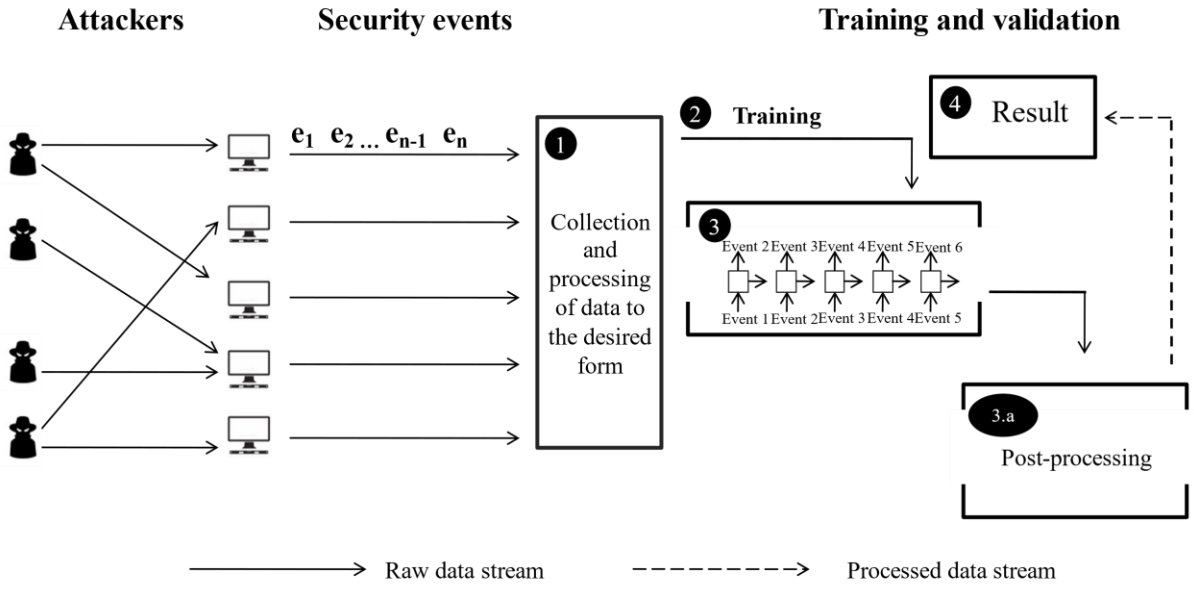

Figure 1: The data processing scheme

During the dataset mining, profile concepts were used: B-profile and M-profile.

B-profile (Benign) – encapsulation of user behavior using various methods of machine learning and statistical analysis (such as K-Means, SVM, Random Forest, and J48).

Encapsulated functions are the size distributions of protocol packets, the number of packets per thread, certain payload structures, the payload size, and the time-division request for the protocol.

The model consists of the following stages:

1. Collecting the network activity data, system logs, and event logs.
2. Processing the preliminary data and bringing them to the required form.
3. Training and testing of the neural network.
4. Analyzing results.

During the simulation, the following protocols were used: HTTP, HTTPS, SMTP, POP3, IMAP, SSH, and FTP. According to the results of the frequency analysis of the data, it was concluded that the bulk of the traffic is represented by HTTP and HTTPS packets. M-profile (Malignant) – an attempt to describe the attack scenario unambiguously. In the simplest case, people can interpret these profiles and then execute them. Ideally, stand-alone agents, together with compilers, will be used to interpret and execute these scenarios. Six different scenarios of attack implementation were considered:

- In the case of network infiltration, a harmful file was sent through email to the target. Once the vulnerability on the target's computer was exploited, the backdoor was activated. This allowed the attacker to scan the internal network using the target's computer and search for any potential new mailboxes.
- In this scenario, an attack known as denial of service HTTP was implemented using Slowloris and LOIC. These tools are capable of making web servers completely inaccessible through one attack system. Slowloris initiates a full TCP connection with a remote server and keeps the connection open by sending valid, incomplete HTTP requests at specific intervals. This is done to avoid closing sockets. As web servers have a limited capacity to serve connections, it becomes a matter of time before all sockets are utilized, and no further connections can be accepted.
- In this scenario, the Damn Vulnerable Web App (DVWA) was utilized to test security analysis skills. DVWA is designed specifically for security professionals. The initial step involves crawling the website using a web application vulnerability scanner, followed by executing different types of web-based attacks, which may include SQL injection, system command injections, and unprotected file upload capability. To detect software vulnerabilities against SQL injections, there are multiple methods such as functional testing (black/white box), phasing, static, dynamic, and manual analysis of the source code. In addition, WAF (web application firewall) is commonly used alongside vulnerability scanning in software applications. WAF offers two security models: signature-based and rule-based, each with its own advantages and disadvantages. However, both models are unable to detect zero-day threats, which means that the attack vector cannot be fully covered [27-30]. Therefore, typical approaches to protect against SQL injection attacks may not provide a sufficient level of security due to low identification accuracy and speed. To counteract known and zero-day attacks, many technologies are being developed based on data mining techniques and the use of neural networks to protect computer systems and networks effectively.
- One common type of cyber attack is a brute-force attack. In this type of attack, hackers try out various combinations of usernames and passwords in order to break into a user's account. There are many tools available for conducting a brute-force attack, including Hydra, Medusa, Ncrack, Metasploit, and Nmap NSE modules. Additionally, there are tools for cracking password hashes, such as hashcat and hashpump. One particularly effective tool is the Python program Patator, which is both flexible and multi-threaded. Patator is able to save the results of each attack in a separate log file, making it easy to review and process later on. In our own testing, we used a password list with 90 million different words.
- Recent attacks have exploited known vulnerabilities, which are severe flaws that can affect millions of servers or victims. These vulnerabilities can be repeatedly exploited and often takes several months to patch all the vulnerable software code. One of the most well-known vulnerabilities in recent years is Heartbleed.

Details of the attacks which were identified, and the means used to implement them are presented in Table 1.

Table 1
Attacks performed on target systems

Type attacks	Tools used
Bruteforce attack	FTP – Patator
	SSH – Patator
DoS attack	Hulk, GoldenEye,
	Slowloris, Slowhttptest
DoS attack	Heartleech
Web attack	Damn Vulnerable Web App (DVWA)
	In-house Selenium Framework (XSS and Brute Force)
Infiltration attack	First level: Dropbox download in a windows machine
	Second Level: Nmap and portscan
Botnet attack	Ares (developed by Python): remote shell, file upload /
	download, capturing screenshots and key logging
DDoS + PortScan	Low Orbit Ion Canon (LOIC) for UDP, TCP, or HTTP
	requests

3. The main results of the research

There are two approaches to analyzing network attacks: one is based on analyzing network activity, and the other is based on analyzing packet content. In this study, we have focused on an approach based on the analysis of network activity. The analysis of network activity was performed using the specialized software CICFlowMeter. CICFlowMeter generates bi-directional streams, where sending the first packet determines the path to the destination source and back to the source system and allows you to get over 80 statistical network traffic attributes. For modeling goals, 67 parameters of network traffic in each thread were identified.

While preparing the data, a new Label attribute was included to distinguish whether a thread represents a specific attack or the regular functioning of information services. As a result, all data was categorized with the following values: Benign, FTP-BruteForce, SSH-Bruteforce, DoS-GoldenEye, DoS-Slowloris, DoS-SlowHTTPTest, DoS-Hulk, DDoS attacks – LOIC-HTTP, DDoS-LOIC-UDP, DDOS-HOIC, Brute Force –Web, Brute Force –XSS, Infiltration, and Bot.

We synthesized the neural network model based on a multilayer Rumelhart perceptron, which is a special case of Rosenblatt perceptron, where weights of neurons are adjusted by back-propagating error correction. The use of more than one layer (usually two or three) is a feature of the approach [24: 31].

The multilayer perceptron developed by Rumelhart served as the foundation for the synthesis of the neural network model. Rumelhart's multilayer perceptron is a particular instance of the Rosenblatt perceptron in which the error-reversing algorithm modifies the weight coefficients of the neurons. The existence of many layers (often two or three layers) is uncommon [24; 31]. The Rosenblatt perceptron-based neural network divides input vectors into the classes 0 and 1, respectively. The input array X and the target array Y, which designate each of the input vectors to one of the two classes, are the two arrays that make up the training sequence.

Input, output, and hidden layers were used in the investigation. The two stages of neural network back-propagation activities are forward and back-propagation. The input pattern is applied to the input layer during the forward propagation stage, and its impact spreads across the network layer by layer until the output value is attained. The error signal for each of the output nodes is calculated after the actual and predicted output values of the network have been compared. The output errors are sent back from the output layer to each node in the hidden (inner) layer, which influences the output layer, as all hidden nodes have more or less contributed to the discovered mistakes in the output layer. The error signal is used to determine the relative contribution of each node to the overall error. This process is repeated layer by layer until all nodes receive an error signal.

The weighting values for each connection are updated using the error data after we have identified the error signal for each node, and this process continues until the network reaches a point where all

training schemes can be encoded. Using a method known as the delta rule or gradient descent, the backpropagation algorithm looks for the least value of the error function in the weight space. For the training task, weights that reduce the error function are thought to be the best option [13; 14].

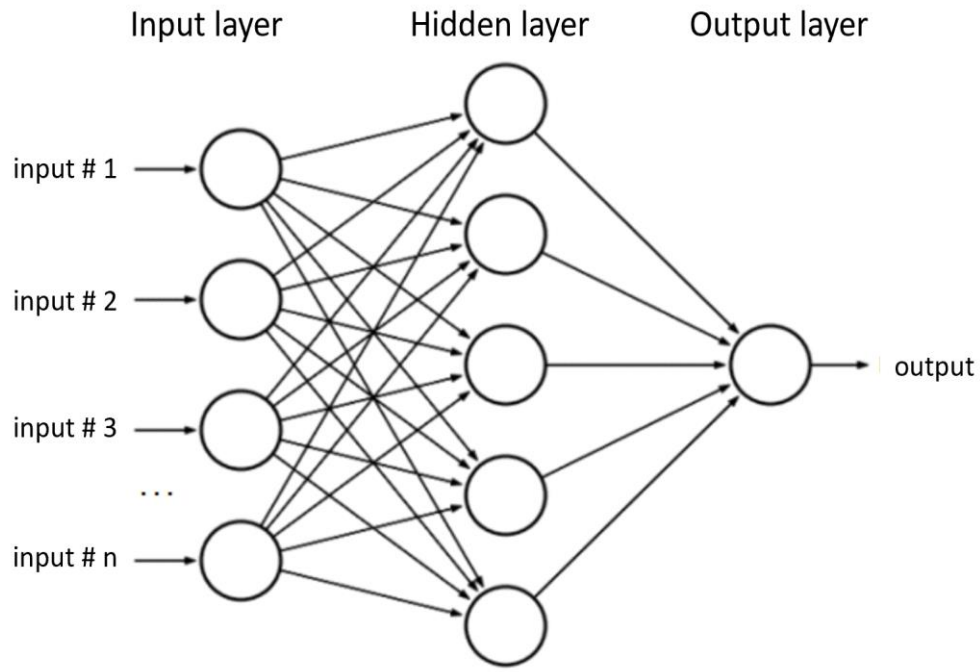

Figure 2: Neural network model

When training, if a certain pattern is fed into the input layer, the weighted sum of the inputs to the jth node in the hidden layer is determined using the following formula:

$$\mathrm{Net}_j = \sum w_{ij} x_j + \theta_j, \tag{1}$$

(1) determines a neuron's total input. θ_j, a weighted shift node with a constant output value of 1, is one example. For each neuron in the hidden and output layers, the shift node functions as a "pseudo-input" and is used to address issues when the value of the input pattern is zero. Without a shift node, the neural network may be trained if any input pattern contains zero values.

The action Net_j potential value is supplied to the appropriate activation function, which uses it to determine whether or not to activate a neuron. The output of the neuron is determined by the value of the activation function, which also serves as the input for the corresponding neurons in the subsequent layers.

For the back-propagation algorithm to work, the activation function must be differentiable. Thus, a commonly used function is the sigmoid equation.

$$O_j = x_k = \frac{1}{1 + e^{-Net_j}}, \tag{2}$$

There could be applied other types of functions for example hyperbolic. (1) and (2) are used to determine the initial value of the node k in the output layer.

The synthesis of the model was based on the creation of its software that implements training and testing of the model. The basic mathematical algorithms used to normalize the data and the training of the model were performed using the Weka API (Application Programming Interface). Weka is open-source software released under the GNU General Public License and contains a set of machine-learning algorithms for data mining tasks. It contains tools for data preparation, classification, regression, clustering, association extraction rules, and visualization. Weka contains API, which is written in Java and implements existing algorithms for training with minimal settings. The final training and validation module was written in the Java programming language.

To identify the event with a given accuracy, it is required that the relative error does not exceed 4%. Due to a large amount of training data, it was divided into several blocks according to the type of attack and it was decided to synthesize a separate neural network model to identify each type of attack.

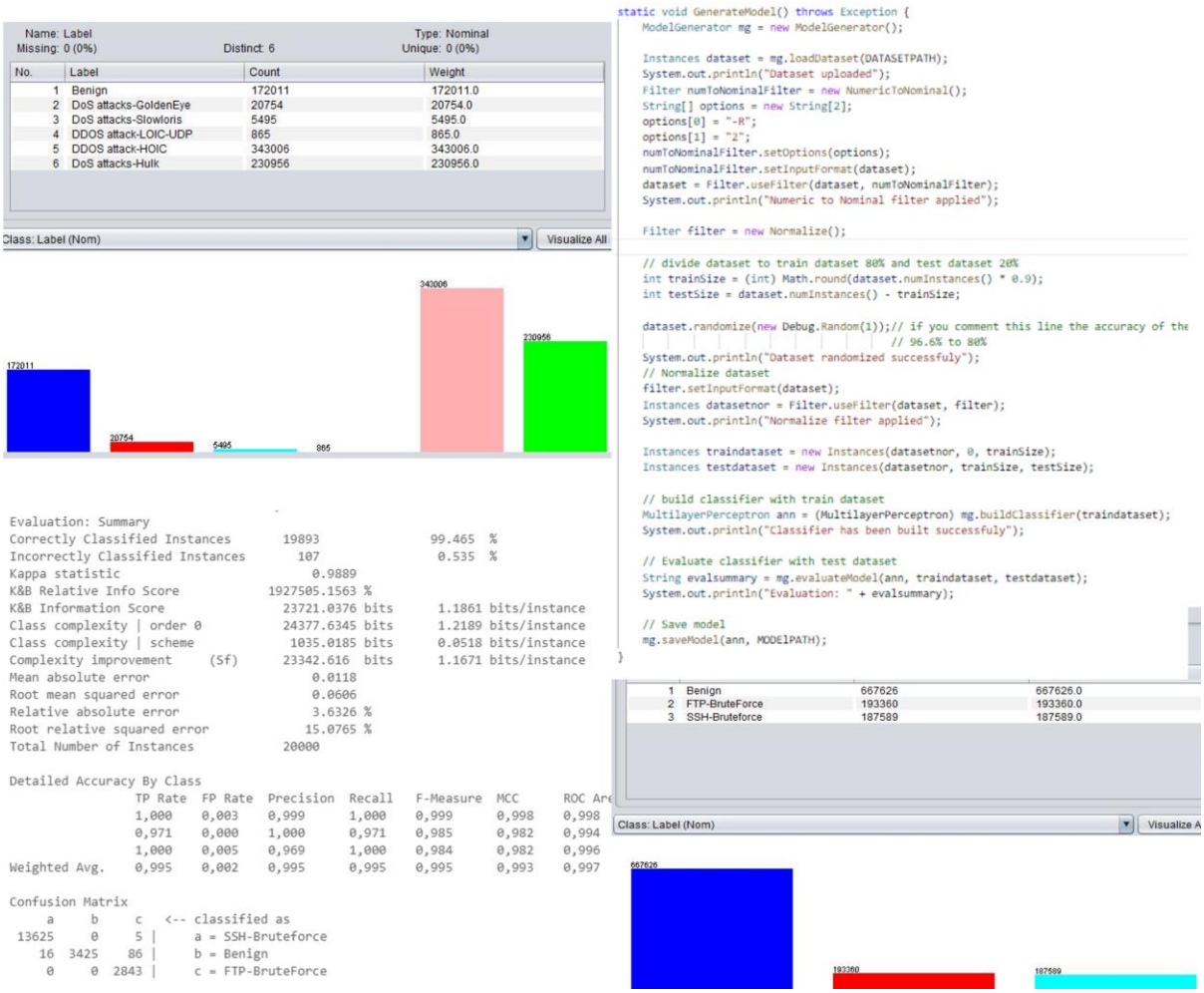

Figure 3: Model learning process

A special procedure with the following algorithm was created to classify attacks using the synthesized model:

1. Loading training data.
2. Determination of nominal values from numerical ones.
3. The neural network represents numerical values in a certain relationship, weights are adjusted from the value of the magnitude. During the data normalization process, dependencies between certain attributes can be misinterpreted and confuse the learning process. For critical numeric attributes such as the port value (1-65565), you need to modify the presentation. To solve this issue the formatting of numeric values to nominal is used.
4. Data normalization.
5. Data normalization in neural networks is the process of optimizing the values of a dataset for numeric types from a large range into the range of values from 0 to 1 while maintaining proportionality. Normalized values significantly increase the learning rate of the model, but do not violate the correctness of its training.
6. The classifier matches the output value of the neural network to the type of attack identified
7. Validators describe the network output classes used to display the classification value to the user.
8. Template classification based on the received response from the neural network.
9. Determination of the network error.

If transmitted data to the classifier contains match patterns, then you can calculate the relative error of the network.

To ensure accuracy in endpoint identification, our training, validation, and test data must come from separate machines. Our model undergoes 100 epochs of training, with regular validation to assess its performance. We select the model that shows the best performance for validation data. Detailed information on assessing the adequacy of the resulting model when presented with a test subset of data is presented in Table. 2. This sample reflects the different types of attacks and their relative error of identification when applied to a synthesized model.

Table 2
Test results of the model

Attack name	Relative error, %
FTP- Bruteforce	0.15
SSH- Bruteforce	0.10
Dos attack GoldenEye	97.50
Dos attack Slowloris	98.96
Dos attack LOIC-UDP	100.00
Dos attack HOIC	0.00
Dos attack HULK	0.00
HTTP Benign	80.40
Infiltration	100.00
Botnet	0.004
Inf-Bot Benign	0.00

4. Conclusions

Analysis of the received results shows that relative identification error when test samples are presented to the synthesized model varies significantly for different types of network attacks. As Table 2 shows, such types of DoS attacks as GoldenEye, Slowloris, LOIC-UDP, Infiltration, and the HTTP Benign cannot be identified by the model. However, in general, further research of the opportunity to use this type of neural network for solving network attack identification issues are rather promising. Upon reaching acceptable results, the accuracy of the identification model will allow not only to identify network attacks but also to perform a network security event forecast based on retrospective data accumulated in the information system over a period and given to the neural network model for training.

5. Reference

[1] P. Chen, L. Desmet, C. Huygens, A study on advanced persistent threats, in: IFIP International Conference on Communications and Multimedia Security, Portugal, 2014, pp 63–72.

[2] G. Stringhini, O. Thonnard, That ain't you: Blocking spearphishing through behavioural modelling. In International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment (DIMVA), 2015.

[3] The White-Hat Hacking Machine: Meet Mayhem, winner of the DARPA contest to find and repair software vulnerabilities, pp. 30–35. https://doi.org/10.1109/MSPEC.

[4] R. Mercy, G. Padmavathi, Self-healing AIS with Entropy Based SVM and Bayesian Aggregate Model for the Prediction and Isolation of Malicious Nodes Triggering DoS Attacks in VANET, International Journal of Computer Network and Information Security 15(3) (2023) 90–105. doi:10.5815/ijcnis.2023.03.07

[5] D. E. Denning, An Intrusion Detection Model, in: Proceedings of the Seventh IEEE Symposium on Security and Privacy, 1986, pp. 119–131.

[6] I. Korobiichuk, S. Fedushko, A. Juś, Y. Syerov, Methods of Determining Information Support of Web Community User Personal Data Verification System, Advances in Intelligent Systems and Computing 550 (2017) 144–150. doi: https://doi.org/10.1007/978-3-319-54042-9_13.
[7] K. Scarfone, P. Mell, Guide to Intrusion Detection and Prevention Systems (IDPS), NIST Special Publication on Computer security (2007) 58–69.
[8] S. M. Bellovin, AT&T Lab Res., USA a look back at security problems in the TCP/IP protocol suite, in: 20th Annual Computer Security Applications Conference, USA, 2004, pp. 268–286.
[9] A. Borkar, A. Donode, A. Kumari, A survey on Intrusion Detection System (IDS) and Internal Intrusion Detection and protection system (IIDPS), in: International Conference on Inventive Computing and Informatics (ICICI), Coimbatore, India, 2017, pp 878–880.
[10] M. Azhagiri, A. Rajesh, S. Karthik, Intrusion detection and prevention system: technologies and challenges, International Journal of Applied Engineering Research 10(87) (2015) 1–11.
[11] R. Daş, M. Baykara. A Survey on Potential Applications of Honeypot Technology in Intrusion Detection Systems, International Journal of Computer Networks and Applications 2(5) (2015) 203–208.
[12] Z. Hu, S. Gnatyuk, T. Okhrimenko, V. Kinzeryavyy, M. Iavich, K. Yubuzova, High-Speed Privacy Amplification Method for Deterministic Quantum Cryptography Protocols Using Pairs of Entangled Qutrits, CEUR Workshop Proceedings 2393 (2019). https://ceur-ws.org/Vol-2393/paper_430.pdf.
[13] V. M. Linh, Q. N. Van, K. Jin-young, K. Kwangki, K. Jinsul, Applications of Anomaly Detection Using Deep Learning on Time Series Data, in: 16th Int. Conf. on Dependable, Autonomic and Secure Computing, 2018, pp. 393–396.
[14] P. Chaudhary, Usage of Machine Learning for Intrusion Detection in a Network, International Journal of Computer Networks and Applications 3(6) (2016) 139–145.
[15] Y. Shen, E. Mariconti, P. A. Vervier, G. Stringhini, Tiresias, in: GSAC Conference on Computer and Communications Security, CCS '18, 2018, pp. 592–605.
[16] Zh. Lianbing, Study on Applying the Neural Network, in: Computer Network Security Assessment 2016 Eighth International Conference on Measuring Technology and Mechatronics Automation (ICMTMA), 2016, pp 639–642.
[17] J. Li, Ch. Dong, Research on Network Security Situation Prediction-Oriented Adaptive Learning Neuron, in: Second International Conference on Networks Security, Wireless Communications and Trusted Computing, 2010, Vol. 2, pp 483–485.
[18] E. Chul, R. Shin, Da. Song, R. Moazzezi, Recognizing Functions in Binaries with Neural Networks, in: USENIX Security Symposium Washington, 2015, pp. 611–626.
[19] A. A. Kuznetsov, A. A. Smirnov, D. A. Danilenko, A. Berezovsky, The statistical analysis of network traffic for the intrusion detection and prevention systems, Telecommunications and Radio Engineering 74(1) (2015) 61–78.
[20] A. Menshawy, Deep Learning By Example. A hands-on guide to implementing advanced machine learning algorithms and neural networks, Pact Publishing Ltd., Birmingham, 2018.
[21] N. I. Naumenko, Yu. V. Stasev, A. A. Kuznetsov, Methods of synthesis of signals with prescribed properties, Cybernetics and Systems Analysis 43(3) (2007) 321–326.
[22] S. Duman, K. Kalkan-Cakmakci, M. Egele, W. K. Robertson, E. Kirda, EmailProfiler: Spearphishing Filtering with Header and Stylometric Features of Emails, in: IEEE 40th Annual Computer Software and Applications Conference (COMPSAC), USA, 2016, pp. 121–126.
[23] O. Hubskyi, T. Babenko, L. Myrutenko, O. Oksiiuk, Detection of sql injection attack using neural networks. Advances in Intelligent Systems and Computing, 1265 AISC, 2021, pp. 277–286.
[24] S. Haykin, Neural networks and Learning Machines, 2nd ed., Prentice Hall, Harlow, 2009.
[25] G. Stringhini, T. Holz, B. Stone-Gross, C. Kruegel, G. Vigna, BotMagnifier: Locating Spambots on the Internet, in: Proceedings of the 2011 USENIX Security Symposium, San Francisco, CA, 2011, pp. 427–443.
[26] S. Toliupa, T. Babenko, A. Trush, The building of a security strategy based on the model of game management, in: 4th International Scientific-Practical Conference Problems of Infocommunications. Science and Technology (PIC S&T), Kharkiv, Ukraine, 2017, pp. 103–108.
[27] Multiple Buffer Format String Vulnerabilities in SQL Server. http://www.microsoft.com/technet/security/bulletin/MS01-060.asp.

[28] Taking the monkey work out of pentesting. http://pentestmonkey.net.
[29] The Web Application Security Consortium. SQL Injection. http://projects.webappsec.org/SQL-Injection, last accessed 2020/02/21.
[30] Microsoft SQL Server extended stored procedure vulnerability (technical explanation and exploit code) – SecuriTeam. https://securiteam.com/windowsntfocus/6n0010u0kw.
[31] Artificial Neural Networks Using Multilayer Perceptron. https://www.cse.unsw.edu.au/~cs9417ml/MLP2/index.html.