

Multischeme feedforward artificial neural network architecture for DDoS attack detection

Arif Wirawan Muhammad¹, Cik Feresa Mohd Foozy², Kamaruddin Malik bin Mohammad³

¹Department of Informatics, Insitut Teknologi Telkom Purwokerto, Indonesia

^{1,2,3}Department of Information Security and Web Technology, Universiti Tun Hussein Onn Malaysia (UTHM), Malaysia

Article Info

Article history:

Received Dec 14, 2019

Revised Feb 20, 2020

Accepted Jul 11, 2020

Keywords:

Architecture

Detection

Distributed denial of service

Multischeme

Neural network

ABSTRACT

Distributed denial of service attack classified as a structured attack to deplete server, sourced from various bot computers to form a massive data flow. Distributed denial of service (DDoS) data flows behave as regular data packet flows, so it is challenging to distinguish between the two. Data packet classification to detect DDoS attacks is one solution to prevent DDoS attacks and to maintain server resources maintained. The machine learning method especially artificial neural network (ANN), is one of the effective ways to detect the flow of data packets in a computer network. Based on the research that has carried out, it concluded that ANN with hidden layer architecture that contains neuron twice as neuron on the input layer (2n) produces a stable detection accuracy value on Quasi-Newton, scaled-conjugate and resilient-propagation training functions. Based on the studies conducted, it concluded that ANN architecture sufficiently affected the scaled-conjugate and resilient-propagation training functions, otherwise the Quasi-Newton training function. The best detection accuracy achieved from the experiment is 99.60%, 1.000 recall, 0.988 precision, and 0.993 f-measure using the quasi-newton training function with 6-(12)-2 neural network architecture.

This is an open access article under the [CC BY-SA](#) license.



Corresponding Author:

Arif Wirawan Muhammad,
Department of Informatics,
Insitut Teknologi Telkom Purwokerto,
Jl DI Pandjaitan 128 Banyumas, 53147, Indonesia.
Email: arif@ittelkom-pwt.ac.id

1. INTRODUCTION

Distributed denial of service attacks (DDoS) is a type of attack that has a quite fatal impact on the target server [1]. DDoS attacks originate from a collection of small-sized data packet streams sourced from a large number of bot computers to produce a massive data flow directed at a target server [2]. DDoS attacks are controlled by attackers who coordinate a large number of bot/slave computers intending to shut down and deplete server resources, including CPU, memory, disk storage, and bandwidth, so that legitimate clients cannot access the allocated resources [3]. In general, packet flow patterns from DDoS attacks behave like normal packet data flow, making it difficult for network administrators to distinguish between proper data packet flow and attack data packet flow [4]. DDoS attacks for a long time can cause the server system to lose all of its services [5, 6]. Detection of the packet flow is one of the techniques to prevent DDoS attacks. Machine learning methods can be used as network packet detection techniques utilizing artificial neural network (ANN). DDoS attacks detection using ANN was once carried out by [7] by utilizing sinusoidal functions to extract essential features from network packet flow resulted in 95.56% accuracy. A neural network with resilient-propagation training function, pooled with the collective classifier output method and

the Neyman-Pearson minimization approach for verification and optimization of detection resulted in 97.45% accuracy, has also been used by [8] to detect DDoS constructed on DARPA and KDDCUP99 datasets. Detection of DDoS attacks by utilizing ANN methods other than based on DARPA and KDDCUP99 datasets, which uses darknet packet flow data has been carried out by research [9]. In the study [10], UDP/53 and TCP/80/8080 packet flow used as ANN training inputs that have been previously extracted by the Local Sensitive Hashing (LSH) method. As compared to research [10], which only detects UDP/53 and TC /80/8080 packet data flow, the study [11] includes ICMP packet flow as ANN training input to detect illegal flows in the network. ANN in research [11] trained by utilizing the backpropagation function. Research [12] proves that the neural network method can be used to identify the novel type of DDoS attacks effectively in the Hbase-Hadoop network environments.

Founded on previous study related to the detection of illegal network packet flow using ANN, this research emphases on the neural network training function and hidden layer architecture that used to detect the DDoS attack packets flow, with the ultimate goal is to find out the best combination of training functions and hidden layer architecture used on ANN to identify regular packets flow and DDoS attack packets flow. The DDoS/malicious packet flow dataset published by UNSW-NB15 University of New South Wales [13] and the normal/regular packet flow dataset published by the Laboratory of Telkom Purwokerto Institute of Technology-Indonesia used in this study.

2. RESEARCH METHOD

To achieve the expected results, the study of detection of normal data packet flow and DDoS attacks packet flow on computer networks using a variety of training function schemes and hidden layer architecture schemes on the ANN involves several steps, as presented in Figure 1.

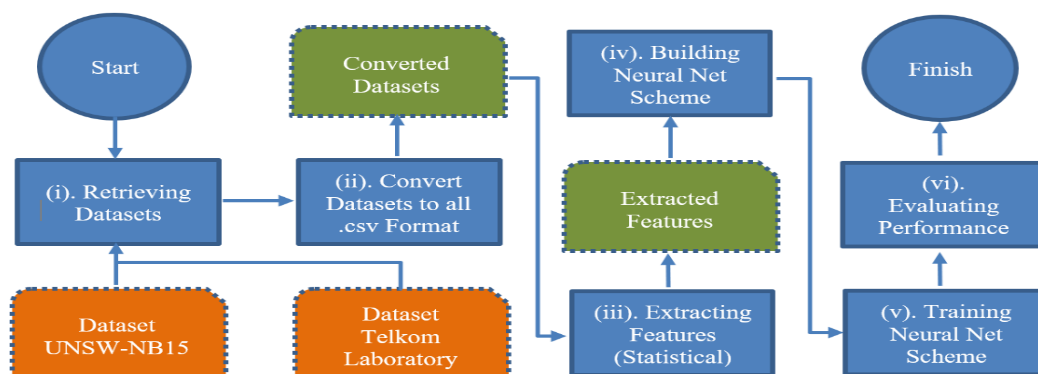
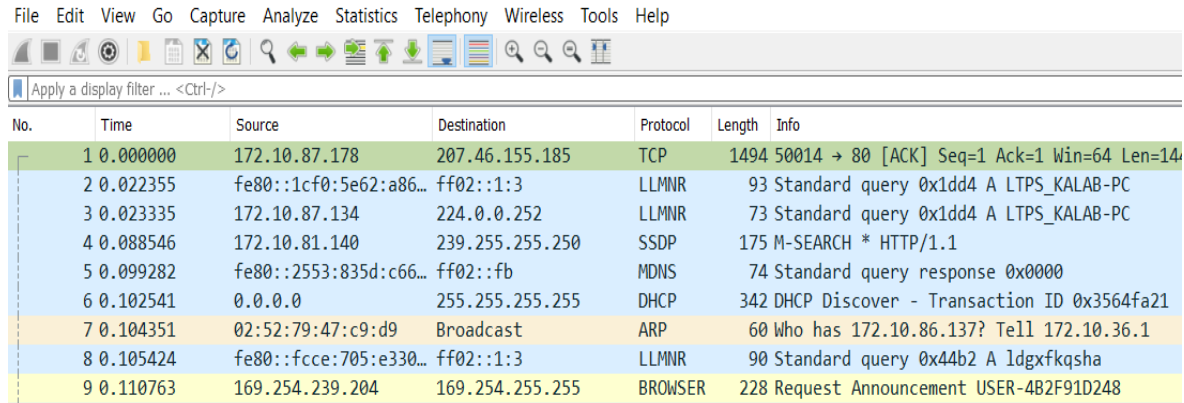


Figure 1. Research method steps

- Retrieving the DDoS attack packet flow dataset from UNSW-NB15 University of New South Wales [13] in the .csv format and the normal/regular packet flow dataset from the Laboratory of Telkom Purwokerto Institute of Technology - Indonesia in the .pcap format as presented in Figure 2.
- Converting the .pcap format packet data stream to the overall .csv format
- Extract data packet flow features based on statistical functions, i.e.:
 - Average packet size (in bytes)
 - Number of packets (in bytes)
 - The variance of packet arrival time intervals (in seconds)
 - The variance of packet size (in bytes)
 - Packet flow speed (in bytes per second)
 - Number of bytes
- Building variation schema on the neural network training functions and the number of neurons in hidden layer architecture.
- Training ANN network architecture schemes with three variations of training functions. First with resilient-propagation, second with quasi-newton, and third with scaled-conjugate).
- Evaluating the performance of ANN training and detection results based on parameters of accuracy, iteration, and mean-squared error (MSE).

In point iv, the accuracy reflects the comparison between the sum of the true positive values (detection of normal packet flow) and the value of true negative (detection of packet flow of DDoS attacks) compared to the total amount of data. On the other hand, MSE is an essential parameter for evaluating the performance of training functions and ANN architecture [14].



No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	172.10.87.178	207.46.155.185	TCP	1494	50014 → 80 [ACK] Seq=1 Ack=1 Win=64 Len=14
2	0.022355	fe80::1cf0:5e62:a86...	ff02::1:3	LLMNR	93	Standard query 0x1dd4 A LTPS_KALAB-PC
3	0.023335	172.10.87.134	224.0.0.252	LLMNR	73	Standard query 0x1dd4 A LTPS_KALAB-PC
4	0.088546	172.10.81.140	239.255.255.250	SSDP	175	M-SEARCH * HTTP/1.1
5	0.099282	fe80::2553:835d:c66...	ff02::fb	MDNS	74	Standard query response 0x0000
6	0.102541	0.0.0.0	255.255.255.255	DHCP	342	DHCP Discover - Transaction ID 0x3564fa21
7	0.104351	02:52:79:47:c9:d9	Broadcast	ARP	60	Who has 172.10.86.137? Tell 172.10.36.1
8	0.105424	fe80::fcce:705:e330...	ff02::1:3	LLMNR	90	Standard query 0x44b2 A ldgxfkqsha
9	0.110763	169.254.239.204	169.254.255.255	BROWSER	228	Request Announcement USER-4B2F91D248

Figure 2. The data packet flow dataset in the pcap format

MSE reflects the absolute error value of the comparison between ANN training outcome patterns and the desired output patterns, while the iteration parameter describes the amount of time required by ANN to achieve convergence and trade-off between time and convergence.

2.1. Network packet flow feature extraction

To detect the normal packet flow and malicious packet flow as presented in point b, the critical step needed is to extract the data packet flow feature from the dataset. The purpose of feature extraction is to produce and quantify the attributes of a dataset so that it can distinguish between one input pattern and another input. In this research, the whole process of extracting and quantizing the data packet flow feature carried out by dividing the fixed time moving window every five seconds. The data packet flow is extracted into six features based on statistical calculations, i.e.:

- Average packet size (in bytes)
Logically, the longer the DDoS data packet flows in the network, the higher the value of the average packet size [15].
- Number of packets (in bytes)
DDoS attacks aim to kill the target server's resources by sending a large number of packets in a specific time lag, resulting in a high accumulation of the number of packets flowing through the network at a particular time [15].
- The variance of packet arrival time intervals (in seconds)
DDoS packets overwrite the network bandwidth massively at a specified time lag, resulting in the arrival time interval value of one packet and the next packet shrinking and approaching zero. The formula for calculating the variance of packet arrival times shown in (1).

$$T_{\text{variance}} = \sqrt{\frac{\sum (T_n - \bar{T})^2}{n}} \quad (1)$$

where T_n is the value of the time the packet was received, and \bar{T} is the value of the overall average packet time received based on the time moving window, while n is the amount of time in the time moving window.

- The variance of packet size (in bytes)
Regular packet data streams produce high packet size variance values. On the other hand, DDoS attack packet flows produce low enough variance values close to zero due to the monotonous size of packets sent by bots to the target server [16]. The formula for calculating package size variances presented in (2).

$$P_{\text{variance}} = \sqrt{\frac{\sum (P_n - \bar{P})^2}{n}} \quad (2)$$

where P_n is the packet size received, and \bar{P} is the average packet size received based on the time moving window, while n is the number of packets in the time moving window.

- e. Packet flow speed (in bytes per second)

The packet flow rate indicates the number of network packets sent from the the attack source to the target server in one time moving window as calculated by (3) [17].

$$P_{\text{rate}} = N_{\text{packet}} \times \frac{1}{T_{\text{end}} - T_{\text{start}}} \quad (3)$$

where N_{packet} is the number of packets received in one time moving window, T_{end} is the time value when the packet was received, and T_{start} is the original packet sent time.

- f. Number of bytes

Is the number of packet size values received in a pause time moving window, which is calculated by (4).

$$Num_{\text{byte}} = \frac{\sum N_{\text{packet}}}{\text{time moving window}} \quad (4)$$

with $\sum N_{\text{packet}}$ is the total size of received packets in one time moving window

2.2. Components of the ANN training function

Some functions/algorithms can use to train ANN networks in batch mode. Functions/algorithms that are widely used by researchers [18, 19]

- Newtonian training algorithm

The Newtonian training function is a training method that can achieve convergence quickly compared to the gradient-conjugate method. However, on the other hand, Newtonian has a reasonably high time complexity because it tends to take a lot of training time to calculate the Hessian matrix in feed-forward ANN [20, 21]. Therefore, there is a derivative of the Newtonian method called quasi-newton (in Matlab: trainlm) which does not necessitate the computation of the Hessian matrix second derivative, because the Hessian matrix automatically updates at each iteration [22].

- Resilient-propagation training algorithm

The resilient-propagation function (in Matlab: trainrp) describes to the algorithm of gradient-descent, which has property removing magnitude effects from the partial derivation of the activation function (e.g linear, sigmoid, binary sigmoid, etc.). In the neural network training mechanism, activation functions partial derivative is used to determine neural network-layer weight update directions, whereas the partial derivation magnitude of the activation function does not have a significant effect on neural network layer weight changes [21].

- Scaled-conjugate training algorithm

The scaled-conjugate function refers to the algorithm of conjugate-gradient, where the algorithm utilizes the adverse direction of the gradient to fit changes in the neural network layer weight. Thus, the conjugate-gradient algorithm has an impact on the number of iterations needed by neural networks to achieve convergence [22].

2.3. ANN layer architecture scheme

Until now, the best neurons arrangement in the hidden layer has not been found in neural network architecture to solve one problem [23], although there is Kolmogorov's concept which states that the amount of neurons in the neural network hidden layer is $2n+1$, where n is the number of input neurons [24]. Based on these reasons, this study conducted several variation schemes on the number of neurons used in the neural network hidden layer, as presented in Table 1. The number of input neurons is six, according to the item extracted from the network packet flow feature. This study uses a neural network architecture with one hidden layer, arguing that a neural network with one hidden layer is sufficient to solve the problem [25]. The more hidden layers tend to have an impact on long training times [7]. On the output side, this study uses two neuron outputs representing the first condition, normal packet flow and the second condition, DDoS attack packet flow. The sample of the first neural network architecture, according to Table 1, shown in Figure 3.

Table 1. The numbers of neuron used in ANN hidden layer

Architecture	Numbers of Neurons at Input Layer	Numbers of Neurons at Hidden Layer	Numbers of Neurons at Output Layer
1 st	6	3	2
2 nd	6	6	2
3 rd	6	12	2
4 th	6	13	2

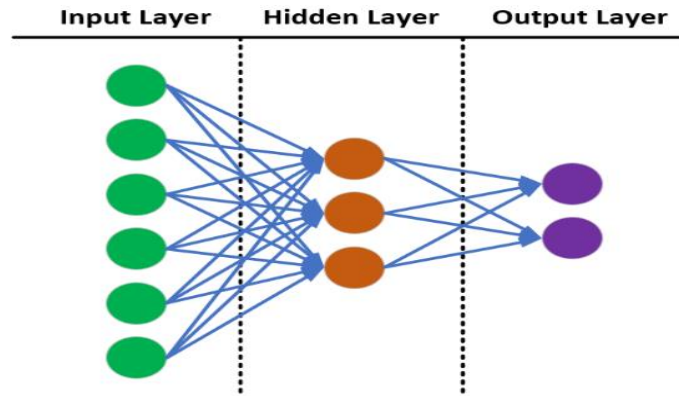


Figure 3. The sample of the first neural network architecture layer (6-(3)-2)

2.4. Evaluation parameters

The performance of the results of experiments conducted in this study will evaluate using three parameters, i.e.:

- Accuracy refers to the comparison among the results of the normal packet flow detection and malicious (DDoS) packet flow, related to the overall packet data.
- MSE, reflects the definite error value from the neural network actual output pattern compared to the desired output pattern.
- Iteration numbers are abstractions of the amount of time required by ANN to achieve convergence.

3. RESULTS AND DISCUSSION

Experiments in this study carried out in the Matlab® R2015b software environment, which runs on a 64-bit Microsoft Windows® 10 platforms. The research involves a normal packet flow dataset and a DDoS attack packet flow with an amount of 1200 data for each packet flow type. Each data consists of six extracted features. By default, the dataset divided into three categories, 70% for training (840-row data), 15% for validation (180-row data), and the remaining 15% for testing (180-row data). The splitting of the dataset into several categories involves the random data sharing function of Matlab® R2015b (in Matlab: dividerand) to prevent the tendency to be biased in the training sample. Parameters used for setting up the training environment are epoch_max=25.10e+3; evaluation-function=MSE; goal-set=0.01, maximum-fail-train=4; gradient-minimum=1.00e-10; mu(μ)=1.00e+10. All result of neural network training includes accuracy, MSE, and iteration presented in Table 2. For simplification, the training results shown for each algorithm/function experiment in this paper are neural networks with 6-(13)-2 layer architecture with Quasi-Newton function training (in Matlab: trainlm) on Figure 4. From the results of the training presented in Figure 4, it can see that the neural network training did not experience overfitting conditions as indicated by the red=test, blue=train, and green=validation lines which decreased the value of MSE smoothly and simultaneously.

Table 2. Training results of neural network scheme

Architecture	Quasi-Newton training function			Resilient-propagation training function			Scaled-conjugate training function		
	Accuracy	MSE	Iteration	Accuracy	MSE	Iteration	Accuracy	MSE	Iteration
1 st (6-3-2)	0.995	0.011	52	0.972	0.042	1388	0.996	0.018	186
2 nd (6-6-2)	0.995	0.009	38	0.979	0.044	746	0.978	0.039	68
3 rd (6-12-2)	0.996	0.010	68	0.996	0.031	1045	0.993	0.030	112
4 th (6-13-2)	0.992	0.010	63	0.989	0.028	451	0.988	0.031	134

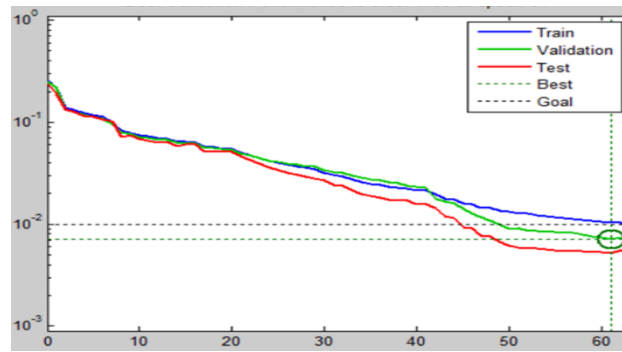


Figure 4. Neural network training result using Quasi-Newton function on architecture 6-(13)-2

3.1. Training accuracy results

Based on the training accuracy results presented in Table 2, it concluded that the Quasi-Newtonian training function provides stable accuracy results on all neural network architectures, with the most significant accuracy value of 0.996 (99.60%) obtained from neural network structure with the hidden layer neurons number as much as $2n$ (n is the number of neuron in input layer), as presented on Table 3.

Table 3. Confusion matrix of 6-12-2 neural network architecture with quasi newton function

Condition	Detected as		Total
	Normal	DDoS	
DDoS	89	1	90
Normal	0	90	90
Total	89	91	180

From Table 3, we can derive :

- Accuracy as $(TP+TN)/(TP+TN+FP+FN)$
 $Accuracy = (89+90)/(89+1+0+90)$
 $=0.996$
- Recall as $(TP)/(TP+FN)$
 $Recall = 89/(89+0)$
 $=1.000$
- Precision as $(TP)/(TP+FP)$
 $Precision = 89/(89+1)$
 $=0.988$
- F-measure $2*Recall*Precision/(Recall+Precision)$
 $=2*1.000*0.988/(1.000+0.988)$
 $=0.993$

The same accuracy results are also obtained by the training function of Resilient-Propagation on neural network architecture with the number of neurons in the hidden layer as much as $2n$ as well as presented in Table 4.

Table 4. Confusion matrix of 6-12-2 neural network architecture with scaled conjugate function

Condition	Detected as		Total
	Normal	DDoS	
DDoS	90	0	90
Normal	1	89	90
Total	91	89	180

From Table 4, we can derive :

- Accuracy as $(TP+TN)/(TP+TN+FP+FN)$
 $Accuracy = (90+89)/(89+1+0+90)$
 $=0.996$
- Recall as $(TP)/(TP+FN)$
 $Recall = 90/(90+1)$
 $=0.989$
- Precision as $(TP)/(TP+FP)$

<i>Precision</i>	$=90/(90+0)$
	$=1.000$
– <i>F-measure</i>	$2*Recall*Precision/(Recall+Precision)$
	$=2*1.000*0.989/(1.000+0.989)$
	$=0.994$

Unlike the highest accuracy in the Scaled-Conjugate training function, although it is 0.996 (99.60%), the accuracy is obtained in neural network architecture with $1/2n$ hidden layer neurons. In general, the resilient-propagation and scaled-conjugate training functions have not been able to provide a stable accuracy value for the entire neural network architecture when compared to the Quasi-Newtonian training function because Quasi-Newton training functions do not need the calculation of the second derivative of the Hessian matrix so that it is faster in achieving convergence.

3.2. Results of mean-squared error training

As presented in Table 2, it can conclude that the Quasi-Newton training function produces a MSE value that is small enough for all neural network architectures (around 0.009 to 0.011) compared to the resilient-propagation and scaled-conjugate functions. In addition, changes in neural network architecture have little impact on the Quasi-Newton training function. Conversely, changes in neural network architecture have a significant effect on the training resilient-propagation and scaled-conjugate model. Increasing the amount of neurons in the neural network hidden layer reduces the resilient-propagation MSE value, whereas in the scaled-conjugate function, increasing the amount of neurons in the neural network hidden layer will raise the value of MSE even though it is not too significant.

3.3. The results of the training iteration

Based on the experiments carried out presented in Figure 4, it found that the neural network with the Quasi-Newton training function provided the least number of iterations compared to the resilient-propagation and scaled-conjugate training functions on all architectures. Quasi-Newton is very fast in achieving convergence, as seen from the small number of iterations <100 . From Table 2, it can be seen that the iteration to achieve the convergence of Quasi-Newton and scaled-conjugate functions is not very much influenced by the type of neural network architecture, while the resilient-propagation training function is less affected by neurons amount in the hidden layer, the higher neurons amount in the neural network hidden layer can accelerate resilient-propagation iteration convergence.

4. CONCLUSION

From the research that has carried out, it found that the neural network architecture of hidden layer neurons is $2n$, $(6-(12)-2)$ (n is the neural network input layer neurons numbers), combined with training function of Quasi-Newton (`trainlm` in Matlab) gives the best accuracy results= 99.60% ; $recall=1.000$; $precision=0.988$; $f\text{-measure}=0.993$. The accuracy value is the same as the accuracy obtained by neural network architecture with $1/2n$ hidden layer configuration $(6-(3)-2)$ trained with scaled-conjugate (`trainscg` in Matlab). The Quasi-Newton training function can produce a reasonably stable accuracy value for all neural network architectures, while resilient-propagation also scaled-conjugate produces accuracy values that are less consistent for all neural network architectures in the experiment. The Quasi-Newton training function is also able to provide faster convergence results compared to scaled-conjugate or resilient-propagation, characterized by the least number of iterations. Judging from the MSE parameters, the Quasi-Newton training function can provide relatively low and consistent results for all neural network architectures. Whereas resilient-propagation also scaled-conjugate produce inverse MSE values. on scaled-conjugate, fewer neurons amount in neural network hidden layer tends to produce elevated MSE values, otherwise, in resilient-propagation, MSE value tends to decrease. With the right combination of training function and neural network architecture, this study achieves higher accuracy (99.60%) to detect DDoS attack compared to other previous research regarding DDoS detection utilizing artificial neural network results that got accuracy under 98.00%.

ACKNOWLEDGEMENTS

The authors would like to thank the Ministry of Education Malaysia for supporting this research under Fundamental Research Grant Scheme Vot No. FRGS/1/2018/ICT04/UTHM/03/4 and partially sponsored by University Tun Hussein Onn Malaysia.

REFERENCES

- [1] M. De Donno, A. Giaretta, N. Dragoni, and A. Spognardi, "A Taxonomy of Distributed Denial of Service Attacks," in *2017 International Conference on Information Society (i-Society)*, Dublin, pp. 100-107, 2017.
- [2] M. A. Naagas, E. L. Mique, T. D. Palaoag, and J. S. Dela Cruz, "Defense-through-deception network security model: Securing university campus network from DOS/DDoS attack," *Bulletin of Electrical Engineering and Informatics*, vol. 7, no. 4, pp. 593-600, 2018.
- [3] A. Saravanan, S. Sathya Bama, S. Kadry, and L. K. Ramasamy, "A new framework to alleviate DDoS vulnerabilities in cloud computing," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 9, no. 5, pp. 4163-4175, Oct 2019.
- [4] O. Osanaiye, K. R. Choo, and M. Dlodlo, "Change-Point Cloud DDoS Detection using Packet Inter-Arrival Time," *2016 8th Computer Science and Electronic Engineering (CEECE)*, Colchester, pp. 204-209, 2016.
- [5] W. Hurst, N. Shone and Q. Monnet, "Predicting the Effects of DDoS Attacks on a Network of Critical Infrastructures," *2015 IEEE International Conference on Computer and Information Technology; Ubiquitous Computing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing*, Liverpool, pp. 1697-1702, 2015.
- [6] O. Osanaiye, K. K. R. Choo, and M. Dlodlo, "Distributed denial of service (DDoS) resilience in cloud: Review and conceptual cloud DDoS mitigation framework," *Journal of Network and Computer Applications*, vol. 67, pp. 147-165, May 2016.
- [7] T. Ishitaki, D. Elmazi, Y. Liu, T. Oda, L. Barolli, and K. Uchida, "Application of Neural Networks for Intrusion Detection in Tor Networks," in *2015 IEEE 29th International Conference on Advanced Information Networking and Applications Workshops*, Gwangju, pp. 67-72, 2015.
- [8] M. K. Rafsanjani and N. Kazeminejad, "Distributed denial of service attacks and detection mechanisms," *Journal of Computational Methods in Sciences and Engineering*, vol. 14, no. 6, pp. 329-345, 2014.
- [9] N. Furutani, T. Ban, J. Nakazato, J. Shimamura, J. Kitazono, and S. Ozawa, "Detection of DDoS Backscatter Based on Traffic Features of Darknet TCP Packets," in *2014 Ninth Asia Joint Conference on Information Security*, Wuhan, pp. 39-43, 2014.
- [10] S. H. A. Ali, S. Ozawa, T. Ban, J. Nakazato and J. Shimamura "A Neural Network Model for Detecting DDoS Attacks Using Darknet Traffic Features," *2016 International Joint Conference on Neural Networks (IJCNN)*, Vancouver, BC, pp. 2979-2985, 2016.
- [11] H. Kaur, S. Behal and K. Kumar, "Characterization and Comparison of Distributed Denial of Service Attack Tools," in *International Conference on Green Computing and Internet of Things (ICGCIoT)*, Noida, pp. 1139-1145, 2015.
- [12] T. Zhao, D. C. T. Lo, and K. Qian, "A neural-network based DDoS detection system using hadoop and HBase," *2015 IEEE 17th International Conference on High Performance Computing and Communications, 2015 IEEE 7th International Symposium on Cyberspace Safety and Security, and 2015 IEEE 12th International Conference on Embedded Software and Systems*, New York, NY, pp. 1326-1331, 2015.
- [13] N. Moustafa and J. Slay, "UNSW-NB15: A comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set)," *2015 Military Communications and Information Systems Conference (MilCIS)*, Canberra, ACT, pp. 1-6, 2015.
- [14] M. Negnevitsky, "Artificial Intelligence A Guide to Intelligent Systems," Edinburgh, Scotland: Addison Wesley, 2011.
- [15] P. Shamsolmoali and M. Zareapoor, "Statistical-based filtering system against DDOS Attacks in Cloud Computing," in *2014 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, New Delhi, pp. 1234-1239, 2014.
- [16] A. Aggarwal and A. Gupta, "Detection of DDoS Attack Using UCLA Dataset on Different Classifiers," *International Journal of Computer Sciences and Engineering*, vol. 3, no. 8, pp. 32-36, 2015.
- [17] N. Moustafa, B. Turnbull, and K. K. R. Choo, "An Ensemble Intrusion Detection Technique based on proposed Statistical Flow Features for Protecting Network Traffic of Internet of Things," in *IEEE Internet of Things Journal*, vol. 6, no. 3, pp. 4815-4830, June 2019.
- [18] C. Chio and D. Freeman, "Machine Learning & Security," 1st ed. California, United States: O'Reilly Media, 2018.
- [19] S. Haykin, "Neural Networks and Learning Machines," vol. 3. New Jersey: Prentice Hall, 2008.
- [20] N. Moustafa, J. Slay, and G. Creech, "Novel Geometric Area Analysis Technique for Anomaly Detection using Trapezoidal Area Estimation on Large-Scale Networks," in *IEEE Transactions on Big Data*, vol. 5, no. 4, pp. 481-494, Dec 2019.
- [21] L. C. Jain, "Recent Advances in Artificial Neural Networks," *Recent Advances in Artificial Neural Networks*, 2018.
- [22] M. Anthony *et al.*, "Neural Network Learning: Theoretical Foundations," Edinburgh, Scotland: Cambridge University Press, 2009.
- [23] Y. H. Hu and J.-N. Hwang, "Handbook of Neural Network Signal Processing," CRC Press, London, United Kingdom, 2002.
- [24] V. E. Ismailov, "On the approximation by neural networks with bounded number of neurons in hidden layers," *Journal of Mathematical Analysis and Applications*, vol. 417, no. 2, pp. 963-969, Sep 2014.
- [25] E. Micheli-Tzanakou, "Supervised and unsupervised Pattern Recognition," London, United Kingdom: CRC Press, 2001.