https://doi.org/10.48009/2_iis_2020_178-185

Issues in Information Systems

Volume 21, Issue 2, pp 178-185, 2020

IMPROVING TRAINING OF NEURAL NETWORK INTRUSION DETECTION SYSTEMS USING FUZZY LOGIC

Loye L. Ray, University of Maryland Global Campus, Loye.Ray@faculty.umgc.edu

ABSTRACT

Today's threats to networks use various techniques that attempt to penetrate protective barriers. This taxes current intrusion detection systems to stay up with these attacks. Training a neural network intrusion detection system is important to detecting dynamic threats facing a network. However, keeping them trained in such a dynamic threat environment can prove challenging. Therefore, finding a fast method of training an IDS is important. This paper shows how the use of a fuzzy logic inference system can improve the training time for neural network intrusion detection systems. Using a combination of both fuzzy inference system and neural network techniques proved successful in reducing the convergence time of intrusion detection systems.

Keywords: neural networks; fuzzy logic, ANFIS; NSL-KDD; Intrusion detection system; training

INTRODUCTION

The era of computers and networks are consistently plagued with dynamically changing threats which makes it difficult for keeping intrusion detection systems (IDS) trained. Neural network IDS (NNIDS) are self-adapting and self-organizing to adapt to these threats (Ray, 2016). Neural network (NN) IDSs use anomaly-based detection and can be trained to detect abnormal behavior. These IDSs consist of multiple layers of neurons to detect intrusions. As threats change, it requires the NNIDS to be retrained to learn the new threats. NNIDS can adapt to new threats if trained effectively (Ray, 2013). A drawback to NNIDSs is that they have to be trained offline and can take time to come back up depending on the amount and complexity of input data. This can open up the network to attackers.

NN algorithms are slow at learning and can take many repetitive checks to discover a threat. These require more efficient learning algorithms (Shihabudheen & Pillai, 2018). Hybrid algorithms that combine different algorithms to improve learning have shown some progress. However, today's IDSs need fast, dynamic learning and be able to be retrained online.

Another issue is the amount of data coming into a NNIDS can be very large. Networks today can quickly generate huge amounts of traffic data that neural networks have to learn about off-line. A NNIDS can get bogged down with too much input data. This can increase the training time or convergence rate to detect a serious threat (Ray & Felch, 2014). Reducing the amount of data can also reduce the convergence rate and increase the chances of not detecting the threat. However, another technique that can work with a NNIDS is needed that can support large data inputs and help train the system online.

The adapting of fuzzy logic techniques to a NNIDS can solve these issues. The use of fuzzy logic techniques can provide a way to improve the IDS learning time. Fuzzy logic is widely used to handle variable data that may equate to a variance in a threat. It can handle the large amounts of data, allow online dynamic learning of new threats, and combines the learning stages to use single training pass process (Shahparast & Mansoori, 2019). This paper will describe fuzzy logic and introduce the use of an adaptive neuro-fuzzy inference system (ANFIS) that combines a neural network and fuzzing inference system. A comparison is done using MATLAB to compare convergence rates of an NNIDS using a hybrid algorithm to that of an ANFIS. This paper will show the importance of combining fuzzy logic with neural networks to form a neuro-fuzzy inference system IDS to overcome the issues described earlier.

NEURO-FUZZY

Fuzzy Logic Defined

The difficulty with training neural networks is how to identify complex patterns or behavior variations. This is where fuzzy logic can be introduced to help in speeding up the training process. To handle uncertainty in making decisions, one can use fuzzy logic. Fuzzy logic uses Boolean logic to help in this decision-making process (Hassan, 2013). These fuzzy logic systems work with membership functions (MFs) and rules to analyze input data. A MF represents the amount a value belongs to a fuzzy set represented by a number between 0 and 1. The relationship between an element and a set is important in determining whether an attack or normal traffic is present. Fuzzy logic uses membership to describe the degree a behavior belongs to a set. The closer an event matches a pattern the more it belongs to the set that defines an attack or normal behavior (Hao, Zhang & Chao, 2015). Fuzzy logic allows a partial membership of an element to a set. These fuzzy systems use If-THEN rules for mapping input data to output data (Masdari & Khezri, 2020).

Use of fuzzy logic provides more efficient risk analysis and comprehensive results when dealing with uncertainty or where degrees data may belong (Qassim, Patel & Mohd-Zin, 2014). This is true when trying to detect dynamically changing threats. Fuzzy logic addresses the formal principles of approximate reasoning which can provide greater accuracy in detecting dynamic threats. Sound foundation to imprecision and vagueness – along with varying degrees of truth can help in improving detection capability. However, having this sound foundation is not enough.

Neuro-Fuzzy Inference System (NFIS) IDS

Combining a NN IDS and one of the fuzzy logic techniques creates a neuro-fuzzy inference system (NFIS) IDS. The NFIS IDS is a hybrid intelligent system that combines the advantages of both neural network and fuzzy logic while eliminating the shortcomings of both (Ojha, Abraham & Snasel, 2019). It uses the learning ability of neural networks to determine parameters by processing data that includes fuzzy sets, memberships and rules. The identification of fuzzy rules is the most important part of designing a NFIS IDS. Redundant and non-optimized rule sets degrade performance and usefulness of fuzzy-based IDS. The lowest number of rules used is best to prevent this condition. Therefore, the compactness of these rules is desired to improve the interpretability of the fuzzy operations and reduces the computational costs. To optimize these rules requires an efficient structure. These rules and MFs are determined at the start of the construction of the NFIS IDS. A more efficient structure is used to optimize the number of rules in adaptive techniques that are appropriate for learning parameters that change slowly (Qaddoum, Hines & Iliescu, 2013). Therefore, the number of fuzzy rules need to be small and the IF part of the rule should be short. This also helps in handling complex systems where changes happen fast and may take time to relearn parameters from these changes. So, the shorter the IF statements and low number of rules can speed up learning in an IDS. Motahari-Nezhad and Mazidi (2016) confirms this and suggests using two fuzzy rules and one output.

The NFIS IDS structure is composed of five layers of neurons and four layers of connections. Each layer consists of one or more neurons. A neuron converts a weighted input vector to a desired output vector. The input layer provides a neuron for each input feature being examined. This layer accepts input data into the NFIS IDS. So, for thirteen features, there are thirteen input neurons. The function of the input layer is to represent input values and directly transmit non-fuzzy values to the next layer.

The next layer determines membership values from fuzzing the input features and is called the fuzzification layer. This is to determine how close one is to a certain MF. The Gaussian function is one type used as the membership function for this layer. The membership degree to a fuzzy set of an object defines a function where the universe of discourse is the domain and the interval is 0, 1 is the range. The most common MF is the triangular or trimf.

The third layer (hidden layer) makes associations between the fuzzy rules and input/output variables. It is also called the fuzzy rule layer because it defines all possible fuzzy rules to specify qualitatively how the output parameter is determined for various instances of the input parameters. These are IF-THEN rules to help in making these associations. They determine how close the output MFs match the input data. A fuzzy inference rules matrix is created

from the weights used in the hidden layer. The weights are adjusted between the fuzzy layer and hidden layer and then between the hidden layer and output layers. The output layer determines the fuzzy decision of the NFIS IDS. A weighted average method is used to decide on the various fuzzy outputs. It also performs defuzzification and calculates output variables (Qaddoum, Hines & Iliescu, 2013).

Another characteristic of a NFIS IDS is that the number of neurons in each layer changes during its operation. Also, the connections between these neurons can change based on the learning algorithm used. Thus, neurons and connections may be added in order to optimize the network. One feature unique to the use of a NFIS IDS is its ability to solve the two-class classification problem with a set of two rules. One rule for normal traffic and the other for abnormal traffic. This will be ideal in intrusion detection of malicious activity on a network. An adaptive neuro-fuzzy inference system was used to determine the best learning time as compared to a neural network IDS.

Adaptive Neuro-Fuzzy Inference System (ANFIS)

The adaptive neuro-fuzzy inference system (ANFIS) model employs a fast training or convergence rate. It does this because it provides the capability for fast learning, online adaptability, self-adjusting, and provides a small amount of error and small computational complexity (Shihabudheen & Pillai, 2018). It accomplishes this by combining two or more learning techniques to help it find parameters to achieve a fast convergence rate (Shihabudheen & Pillai, 2018). The ANFIS uses a comparatively small number of adjustable parameters that allow it to tune in real time under nonstationary processed information (Bodyanskiy, Vynokurova, Setlak, Peleshko & Mulesa, 2017). The ANFIS can learn while online and provide fast, dynamic learning of new and unknown threats (Ojha, Abraham & Snasel, 2019). The online and dynamic learning comes from the system adapting its parameter and structure for each sample that it sees (Ojha, Abraham & Snasel, 2019). This helps overcome the issue of training off-line. This model provides a systematic approach to creating fuzzy rules from a select dataset input-output and less reliant on human expertise (Varnamkhasti & Hassan, 2013). Thus, it can easily model nonlinear functions because the MFs can be extracted for the data set used. The ANFIS can learn features in a dataset and adjust itself to reduce error to handle large datasets.

The ANFIS is composed of a five-layer architecture. The first layer fuzzificates the input variables. Fuzzification is the process of changing a scalar value into one that is based on a MF. The second layer deploys T-norm operators to compare the rule antecedent. The third then normalizes the rule strengths. The fourth layer determines the parameters of the rule. The last is the output layer that computes the final result.

An ANFIS performs fuzzy classification by breaking up features into classes. These feature spaces are grouped into regions and controlled by rules (Azar & El-Said, 2013). Fuzzy MFs and cluster weights are optimized. The rules then can be adapted with neural networks (Nemissi, Seridi & Akdag, 2014). Thus, improving the fuzzy part using NN methods.

The ANFIS model uses fuzzy antecedents and consequents parts. Each layer performs a different function that contributes to the learning process. The ANFIS uses a hybrid learning algorithm to improve the learning algorithm. This is so it can better understand the parameters of the ANFIS and provides dynamic learning capability (Shubair, Ramadass & Altyeb, 2014). To solve the issue of handling large data traffic, a hybrid algorithm composed of both least squares estimator and gradient descent method was used. This allows the model to learn from both neural and fuzzy systems. The least squares method uses a forward pass to optimize the parameters of the system with the parameters fixed. This helps the model determine precise parameters related to membership functions. The output error from the forward pass is used to adjust the parameter weights before starting the gradient descent. Then a gradient descent or backward pass is performed to adjust the parameters to an optimal level according to the input parameters. The hybrid algorithm provides an efficient means of training the network and allow the system to use a one pass learning process (Shihabudheen & Pillai, 2018).

In an ANFIS, fuzzy input and rules are used to obtain fuzzy output. The rules and membership functions are strongly interdependent. They provide a way to explain characteristics of a problem and transform into rules that make complex problems into straightforward language one can understand (Souza, Rezende, Guimaraes, Araujo, Batisla, Silva & Araujo, 2019). The ANFIS maps input space into output space by fuzzy rules (Hao, Zhang & Chao, 2015). Fuzzy rules use IF-THEN statements to determine how well information matches a pattern. These rules are gathered into a rule base for helping the system better learn normal and attack traffic. Fuzzy sets deal with approximate rather than

precise information. The advantage of using fuzzy is their interpretability. It uses fuzzy modeling to identify the parameters of the ANFIS to obtain a desired result.

METHODOLOGY

To show a difference in training or convergence rate in an ANFIS IDS model, it was compared against a feed forward neural network IDS model. Each model was trained separately and data collected on the learning time required to train each model. It was noted that both ANFIS and NN IDS nodes, and connections change as data was presented to it. Both models consisted of several inputs with each based on different attributes from normal and attack data. The number of these inputs were the same and fixed. Each model consisted of several layers.

ANFIS Model

For the ANFIS model, the first layer passed information to the next where it then calculated the degree that the input values belong to a given fuzzy membership function. Figure 1 shows how the AFNIS model was built. This second layer represented a fuzzy quantization of each input variable space. For example, each input neuron can represent small, medium or large fuzzy values. Then different MFs (triangle, Gaussian, etc.) were attached to these neurons. For this research, the triangular MF (trimf) was used because it provided a simpler MF. The purpose was to be able to transfer input values to degrees of membership for which the input belongs. The amount and type of MFs were dynamically modified during the learning process. This allowed the ANFIS IDS to adjust to changes in the input data stream over time while preserving the generalization capabilities of the system.

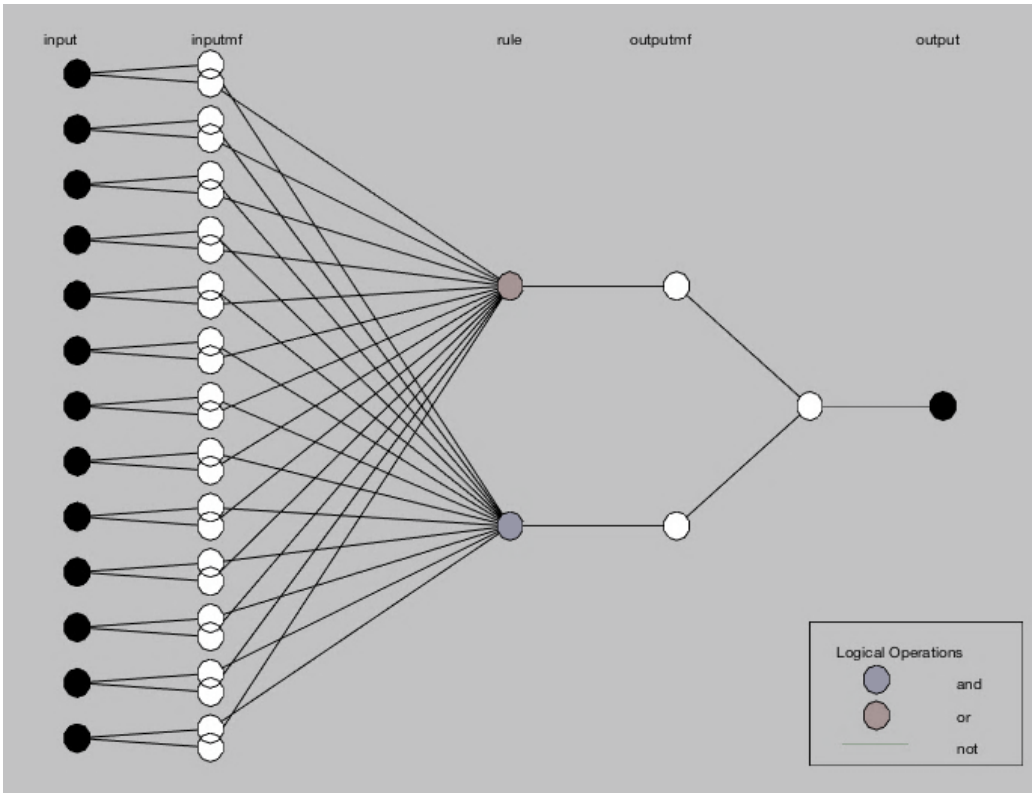

Figure 1. Model of ANFIS used

The third or rule layer represented clusters of input data while hosting modification to the neurons. It contained rule nodes that represent both input and output data defined by weighted vectors. Each rule node was defined by two connection weighted vectors. These were either high or low. The fourth layer determined how well the output membership functions match the input data. It performed as a fuzzy quantization of the output variable. A weighted sum input function and linear function was used by the neurons to calculate the degrees of membership the output vector is associated within the specific input vector in each output MF. The fifth or fuzzy output layer performed defuzzification and calculated precise output values. The output layer provided the final result from the ANFIS model. It used at least two membership functions per input.

Neural Network IDS Model

In the feed forward neural network model, a fixed number of input neurons was used based on the number of features introduced to the IDS (see Figure 2). Then the number of hidden layer neurons was set to a low number and varied to get the best convergence rate. The output neurons were set to one for detection of an attack or normal activity. This was the most efficient number to detect an attack or not. The ANFIS model used the same number of outputs as well. The backpropagation algorithm was used to adjust weights of the connections in the model.

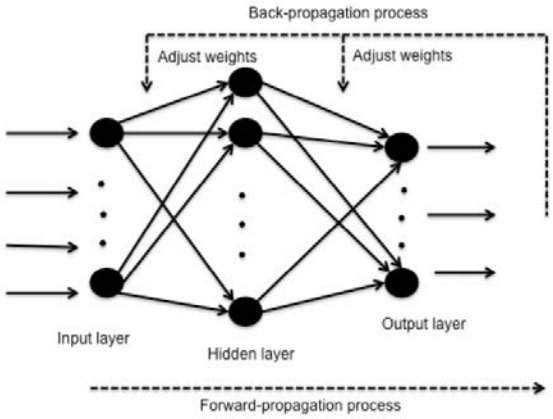

Figure 2. Feed Forward Neural Network Model Used

Training the Models

Both IDS types were trained using the NSL-KDD public dataset. The NSL-KDD dataset is derived from the KDD-98 dataset. The NSL-KDD was selected because it corrects many of the problems with the KDD-98. The KDD-98 contains many duplicated data patterns and errors. The NSL-KDD fixed these issues and provides a better source of test and training data (Dhanabal & Shantharajah, 2015). Real network traffic data is preferred over any static public data if available. However, real data is difficult to acquire due to privacy concerns of its owner. Use of real data may expose an organization to attackers if it got in the wrong hands.

Public datasets such as KDD 98 contains several anomalies such as duplication and non-numeric information (Dhanabal & Shantharajah, 2015; Mukosera, Mpofu & Masaiti, 2014). For the ANFIS and NN IDS to work properly, one needed to correct these anomalies. Therefore, the NSL-KDD data set was used since it corrected many of these abnormalities. Also, since both systems only used numeric data, the dataset symbolic information was converted. Preprocessing data involves cleaning up incomplete data by mapping symbolic value attributes to numeric-valued attributes. Also, the amount of data in these datasets are too large and cost more in computation time to adequately determine an intrusion. This was called the curse of dimensionality. It was caused when computation requirements grow exponentially with the number of variables. The NSL-KDD dataset has 41 variables that can create a complex IDS model. This complexity can increase the time to train an IDS. Thus, the number of features or variables of the

data input need to be reduced. Thus, feature selection was necessary to reduce the number of features necessary to compute intrusions important to the effectiveness. There was also a relationship between the number of input variables and the number of rules in the rule set. In fuzzy inference, an optimal feature set can produce a minimum number of rules.

A limited amount of the dataset was used for training. In this case, 20% of the dataset was used for training (Jing, Bi & Deng, 2016). The NSL-KDD has a 20 percent dataset available but needs to be adjusted for this research. The 20% training set was too big so feature selection was used to reduce the number of features and size of the dataset used. The number of features were reduced from 41 to 13 based on using what was normal behavior. Mukoser, Mpofu and Masaiti (2014) recommended this approach to find the important features needed for creating the fuzzy rules. Microsoft Excel was used for converting the symbolic information and reducing the number of record attributes. Using this on both models made it easier to detect abnormal traffic. This would also reduce the chances of errors. The use of records and concentration of only attributes of interest greatly helps reduce the amount of information to be used. Those features picked included service_type, land, wrong_fragments, urgent, hot, num_failed, num_compromised, rshell, count, srv_count, dst_hst_srv_diff_host_rate, dst_hst_serror_rate, dst_srv_rerror_rate. These were selected because under normal conditions remained low values making detecting attacks easier to find. This was because an attack exhibited a high value.

RESULTS

The MATLAB tool was used to create the ANFIS and NN IDS models. The Neural Network Toolkit was used to build the NN IDS while the Fuzzy Logic Toolkit was used to build the ANFIS model. The ANFIS was composed of 72 neurons, 78 nonlinear parameters and 28 linear parameters. The number of training data pairs was 25192 and the extracted fuzzy rules were 2 rules. The number of input data vector attributes was 13, number of each which has 2 membership functions. The initial error size was set to 0 and the number of epochs to 3. The hybrid learning algorithm was used to train the system. The percent of error in the training was 0.304542. The ANFIS completed training at epoch 2.

For the NN IDS, the number of input neurons was fixed at 13 and the output neuron at 1. The number of hidden layer neurons was varied from 5 to 80 to gauge the effects to convergence rate. For the same input, the NN IDS took 89 to 400 epochs to train. The amount of error varied from 3.6 to 6.4 percent. The results of the training of the ANFIS and NN IDS showed that the ANFIS took less than 2 epochs while the NN IDS took nearly 100 or more epochs. This shows that the ANFIS IDS can quickly train and adapt to new threats.

The ANFIS IDS used if-then rules on the inputs to obtain a single output. There were two if-then rules constructed from the input data. One was for normal traffic detection while the other was for attacks. It relied more on interpretation of the input data (Shubair, Ramadass & Altyeb, 2014). Automatically learning from these rules made the ANFIS IDS ideal for adaptive learning while reducing false positives and negatives.

A significant part of the ANFIS IDS was that all of the nodes in the network were created while learning. This supported its capability of one-pass learning and reduction of convergence rate. Also, it used triangular functions in learning that provided better online generation and accurate node allocation. The ANFIS IDS used a hybrid learning technique to fine tune the parameters. This supported continuous learning of new data in the ANFIS IDS. The ANFIS IDS did a single-pass training making it more adaptable and easier for on-line training. However, the learning procedure was faster when using linear equations for calculating activation of rule nodes (Qaddoum, Hines & Iliescu, 2013). Another advantage was that the ANFIS IDS used a fast-adaptive local learning technique to reduce adding many new neurons or connections as new data was introduced (Qaddoum, Hines & Iliescu, 2013).

CONCLUSION

This research showed that an ANFIS IDS can outperform an NN IDS in learning new threats. Its adaptability and fast learning capability were proven to be superior to the NN IDS. Keeping the number of IF-THEN rules low helped to achieve this result. Also reducing the number of input variables can produce a significant improvement in an IDS learning and training time. This research showed that training of an ANFIS or NN IDS systems can be negatively affected by the number of input variables used. However, using feature reduction proved useful in overcoming this

issue. This report showed that reducing the number of input variables helped improve the training time for both systems.

FUTURE WORK

The use of different data sources is needed for future IDS development. To do so can provide a better and truer output to today's dynamic threat environments. Future testing should use real data to get more exact results in the use of fuzzy logic in training an NN IDS. Also training IDSs with public datasets may not give the same results as with real data due to the datasets being outdated. Future research needs to use real traffic in order to be more practical if deployed to combat today's attacks. The use of a honeynet system can be a good source of current intrusions to train an IDS. Also, further work is needed in seeing how ANFIS systems, can handle detecting different attacks vectors. Modifying the output to defect various types of attacks could be of great benefit in combating them. Lastly, using a larger dataset needs to be used to test the training time since most networks could have millions of events occur in a short time frame.

REFERENCES

Azar, A.T. and El-Said, S.A. (2013). Superior neuro-fuzzy classification system, *Neural Computing and Applications*, *23*, 55-72.

Bodyanskiy, Y., Vynokurova, O., Setlak, G., Peleshko, D. & Mulesa, P. (2017). Adaptive multivariate hybrid neuro-fuzzy system and its on-board fast learning, *Neurocomputing*, *230*, 409-416.

Dahanabal, L. and Shantharajah, S.P. (2015, June). A study on NSL-KDD dataset for intrusion detection system based on classification algorithms, *International Journal of Advanced Research in Computer and Communications Engineering, 4*(6), 446-452.

Hao, Z., Zhang, Z. and Chao, H. (2015). A cluster-based fuzzy fusion algorithm for event detection in heterogeneous wireless sensor networks, *Journal of Sensors*, 1-11.

Hassan, M.M.M. (2013, March). Current studies on intrusion detection system, genetic algorithm and fuzzy logic, *International Journal of Distributed and Parallel Systems, 4*(2), 35-47.

Jing, X., Bi, Y. and Deng, H. (2016, July). An innovation two-stage fuzzy knn-dst classifier for unknown intrusion detection, *The International Arab Journal of Information Technology, 13*(4), 359-366.

Masdari, M., & Khezri, H. (2020). A survey and taxonomy of the fuzzy signature-based intrusion detection system, *Applied Soft Computing Journal*, *92*, 1-19.

Motahari-Nezhad, M. & Mazidi, M. (2016). An adaptive neuro-fuzzy inference system (AFNIS) model for prediction of thermal contact conductance exhaust value and its seat, *Applied Thermal Engineering*, *105*, 613-621.

Mukosera, M., Mpofu, T.P. and Masaiti, B. (2014, June). Analysis of NSL-KDD dataset for fuzzy based intrusion detection system, *International Journal of Science and Research*, *3*(6), 1479-1482.

Nemissi, M., Seridi, H., & Akdag, H. (2014). One-against-all and one-against-one based neuro-fuzzy classifiers. *Journal of Intelligent & Fuzzy Systems*, *26*, 2661-2670.

Qaddoum, K., Hines, E.L. and Iliescu, D.D. (2013). Yield prediction for tomato greenhouse using EFuNN, *ISRN Artificial Intelligence, 2013*, 1-9.

Qassim, Q., Patel, A. and Mohd-Zin, A. (2014, September). Strategy to reduce false alarms in intrusion detection and prevention systems, *The International Arab Journal of Information Technology, 11*(5), 500-506.

Ojha, V., Abraham, A, & Snasel, V. (2019). Heuristic design of fuzzy inference systems: A review of three decades of research, *Engineering Applications of Artificial Intelligence*, *85*, 845-864.

Ray, L.L. (2013). Training and testing anomaly-based neural network intrusion detection systems, *International Journal of Information Security Science, 2*(2), 57-63.

Ray, L.L. & Felch, H. (2014). Improving performance and convergence rates in multi-layer feed forward neural network intrusion detection systems: A review of the literature, *International Journal of Strategic Information Technology and Applications, 5*(3), 24-36.

Ray, L.L. (2016). Challenges to multi-layer feed forward neural networks in intrusion detection, *Issues in Information Systems, 17*(1), 89-98.

Shahparast, H., & Mansoori, E.G. (2019). Developing an outline general type-2 fuzzy classifier using evolving type-1 rules. *International Journal of Approximate Reasoning, 113*, 336-353.

Shihabudheen, K.V. & Pillai, G.N. (2018). Recent advances in neuro-fuzzy system: A survey, *Knowledge-based Systems, 152*, 136-162.

Shubair, A., Ramadass, S. and Altyeb, A.A. (2014). kENFIS: kNN-based evolving neuro-fuzzy inference system for computer worm's detection, *Journal of Intelligent and Fuzzy Systems, 26*, 1893-1908.

Souza, P.V., Rezende, T.S., Guimaraes, A.J., Araujo, V.S., Batista, L.O., Silva, G.A., & Araujo, V.J.S. (2019). Evolving fuzzy neural networks to aid in the construction of systems specialists in cyber-attacks, *Journal of Intelligent & Fuzzy Systems, 36*, 6743-6763.

Varnamkhasti, M.J. and Hassan, N. (2013). A hybrid of adaptive neuro-fuzzy inference system and genetic algorithm, *Journal of Intelligent and Fuzzy Systems, 25*, 793-796.