IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received January 10, 2018, accepted March 18, 2018, date of publication March 27, 2018, date of current version April 25, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2820092

A New Intrusion Detection System Based on Fast Learning Network and Particle Swarm Optimization

MOHAMMED HASAN ALI[1], BAHAA ABBAS DAWOOD AL MOHAMMED[2], ALYANI ISMAIL[2], (Member, IEEE), AND MOHAMAD FADLI ZOLKIPLI[1]

[1]Faculty of Computer Systems and Software Engineering, University Malaysia Pahang, Malaysia 26300
[2]Department of Computer and Communication Systems Engineering, Faculty of Engineering, Universiti Putra Malaysia, Malaysia 43400

Corresponding author: Mohammed Hasan Ali (mh180250@gmail.com)

ABSTRACT Supervised intrusion detection system is a system that has the capability of learning from examples about the previous attacks to detect new attacks. Using artificial neural network (ANN)-based intrusion detection is promising for reducing the number of false negative or false positives, because ANN has the capability of learning from actual examples. In this paper, a developed learning model for fast learning network (FLN) based on particle swarm optimization (PSO) has been proposed and named as PSO-FLN. The model has been applied to the problem of intrusion detection and validated based on the famous dataset KDD99. Our developed model has been compared against a wide range of meta-heuristic algorithms for training extreme learning machine and FLN classifier. PSO-FLN has outperformed other learning approaches in the testing accuracy of the learning.

INDEX TERMS Fast learning network, KDD Cup 99, intrusion detection system, particle swarm optimization.

I. INTRODUCTION

In recent years, computer network security is a major concern of computer society due to the development of technologies and internet services at a rapid pace. Developments in computer technology have enabled various new possibilities, including the ability to remotely manage and control systems, as well opening up a gateway to a multitude of information through online sources. Organizational level cyber security has consequently become a chief concern, Goodarzi *et al.* [1] explored the problems faced by organizations in keeping their information protected, available and reliable. This has created the motivation for keeping systems secured from any external system, program, or person aiming at breaking the security line of the network. There are many tools and applications developed to increase the security of the environments like systems, networks and computers. Intrusion Detection System (IDS) is one of that tools that tries to protect the systems from an intruder. IDS monitors the single machine or computer network for intruder [2]. It is useful not only in detecting successful intrusions, but also in monitoring attempts to break security, which provides important information for timely counter-measures [3].

The initial proposal to use intrusion detection in an attempt to address misuses and networking attacks in computers, was put forth by Denning [4] in 1987. The process is implemented by an intrusion detection system. Presently such systems are widely available with variety [5], points out the general ineffectiveness and lack of sufficiency provided by the present commercially available systems, this brings to light the need for ongoing research on more dynamic intrusion detection systems. In order to execute the process of intrusion detection, there is a need to identify ongoing or attempted intrusions or attacks on the system or network, this identification data include data collection, behavior classification, data reduction, and lastly reporting and response, this is referred to, as ID [6].

The IDS attempted to determine whether monitored user activity or network traffic is malicious. If a malicious attack is detected, an alarm would be generated. Various different techniques are available for IDSs' to distinguish an attack, such as anomaly detection or signatures of attack, Green *et al.* [7] also point out that the success of IDS depends upon these techniques. One amongst the principal factors governing the efficacy of the IDS is the quality of the feature construction and feature selection algorithm. In order to improve the overall efficiency of the IDS, a drop in the number of applicable traffic features without incurring any adverse effects on classification accuracy is required.

In recent times, we have seen an exponentially great increase in the employment of Artificial Intelligence (AI) in a tremendously large and vast number of fields, such as; computer vision, robotics, control, communication and various engineering fields. AI combined of several sub fields such as neural network, evolutionary searching, expert systems, fuzzy systems, etc. Although a lot of researchers prefer AI models with interpretability aspects such as heuristically knowledge building based models like fuzzy systems, artificial neural networks ANN, which had no explicit interpretability aspect is considered as more effective AI models when learning scheme is feasible. This is due to the power of capturing knowledge through examples provided to such models. This has created a strong motivation to researchers for building supervised learning models to predict intrusion attacks based on collected data set of examples of various attacks. There exists a very large number of methods, most of which have been used for different intrusion-detection models to perform a diverse set of important tasks, some of these methods include; Machine learning based, Hybrid ANN based and/or integrated techniques. Additionally, as presented by Kiranyaz *et al.* [8], there are hybrid data mining schemes, hierarchical hybrid intelligent system models, and ensemble learning approaches all of which have gained popularity in the works reviewed. The remainder of the present work is arranged as such; we start in section 2 with related work. Section 3 talk about the data set KDD. Section 4 formulation problem. Section 5 developed methodology. Section 6 results and discussion. The conclusion and summary in this work in section 7.

II. RELATED WORK

Artificial Neural Networks (ANNs), from input patterns, it can be approximate complex nonlinear mappings directly, and has been used in a lot of applications with great success [7]. Artificial Neural Networks (ANNs), given their ability to approximate complex nonlinear mappings directly from input patterns, have been frequently used in a variety of applications with great success [9]. Based on gradient descent algorithms training samples would be used to define the free parameters of ANNs. Moreover, this reason for brings some issues related to its local minima and the learning process relatively became slow. Owing to these shortages, also train ANNs could take much more time and have a suboptimal solution [8]. For solving the above problems, it has been a hot topic to reduce the computing iterations and simultaneously decrease the training time [9]–[11]

In order to address the aforementioned problems, Huang *et al.* [12] propose the use of a new artificial neural network, known as an Extreme Learning Machine (ELM). ELM is defined as new learning approach for Single Hidden Layer Feedforward Neural Network (SLFN), where random value generation is used for the input weights and the bias of hidden nodes without tuning, and where the output weights are determined analytically.

Extreme learning machine as explored by (Huang et al., 2004), avoids several disadvantages of gradient descent-based learning algorithm for SLFNs. Research on the approximation abilities of Feed-Forward Neural Networks (FFNN's) focuses on two primary features: universal approximation on compact input sets and approximation in a finite set of training samples [12]. Some general advantages of ELM algorithms are; simple and robust implementation, tendency to converge with the shortest training error, and smallest norm of weights, and generally good performance, with extremely fast running. These amongst other help differentiate ELM from the other SLFN algorithms.

The ELM algorithm is based on three steps training; firstly, assigning random weights in the input-hidden layer, secondly, calculating the output hidden layer matrix, and thirdly, calculating the output layer weights based on Moore-Penrose equation [11]. Based on the idea of ELM, Li *et al.* [13] proposed a novel Fast Learning Network (FLN). The FLN is a Double Parallel Forward Neural Network (DPFNN) [14], which is essentially a parallel connection of a multilayer FFNN, and an SLFN. The re-coded external information from the hidden nodes, along with the external information itself directly from the input nodes is fed into the output nodes of the DFNN's. Input weights as well as hidden layer biases are generated in a random manner for FLN's, but where an analytical approach, based on a least squares method is used to determine the weights of values for the connection between the output layer and the input layer and the weights of values for connecting the output node and the input. If a comparison is made between relating methods FLN, is capable of reaching a good general high speed performance, with impressive stability in most scenarios, whilst running with a smaller number of hidden units.

In order to build an effective and reliable ANN based intrusion-detection system, there is a high need to provide comprehensive data set for teaching the ANN model. Although several data sets exist within the literature for such a knowledge building, there is a significant challenge that needs to be addressed in this respect. More specifically, most of the dataset do not provide enough examples for teaching the models in an explicit way due to the less frequency of some attacks. This has caused a concern on how to rely on the available small examples of data of attacks in order to build generalizable knowledge for AI models to use it in detecting similar non-stored attacks. An example for one common dataset used for training models on intrusion attacks is KDD99.

Although ELM approach of training for both SLFN and FLN is quite easy and provides non-iterated learning for the model it has one important limitation. Actually, it is having an infinite number of degree of freedom to reach a classification result. In other words, there is no one deterministic solution to train an SLFN network with basic ELM training. Assuming that the possible weights of the input-hidden layer connections are potential solution for training ELM, there are certain values of set of solutions with more superiority if the goal is to obtain best knowledge extraction from the data set. We call the process of finding those solutions based upon an extension of ELM a developed ELM. Our goal is to design a learning mechanism based on two factors: the nature of the data set, and the nature of the evaluation measures that are aimed to be used for evaluating the learning mechanism or algorithm.

III. DATA SET KDD99

ANN based intrusion detection has to be trained on selected Dataset. In order to demonstrate the effectiveness of our model, we choose the highest dataset in terms of citation to the literature of intrusion KD99. Furthermore, we present the different issue that is addressed in the literature.

A. OVERVIEW OF KD99

KDD Cup 99 is considered the most accepted research dataset highly appropriate to benchmark performance [17], also notes its use in comparing the effectiveness of various approaches to Network Intrusion. KDD CUP 99 is built based on the data captured in DARPA'98 IDS program [18]. DARPA'98 contains approximately 4GB of compressed raw (binary) tcpdump data. This contains roughly 7 weeks of monitored network traffic. This data can consequently, be managed into about 5 million linking records, each about 100 bytes. KDD training data set consists of approximately 4,900,000 single connection vectors each of which contains 41 features and is labeled as either normal or an attack, [19], the attacks can thereafter be categorized into exactly one of four, as detailed below;

Denial Service of Attack (DoS): DOS is an attack which essentially involves the resources are too busy to handle other requests or the attacker making use of specific resources to an extent that denied access for legitimate users.

User to Root Attack (U2R): It is a form of security exploitation, whereby the attacker would gain access to a normal user account, through conventional means, and thereafter proceed to attempt root access to the system through the exploitation of a vulnerability.

Remote to Local Attack (R2L): this is when an attacker attempts access to a system over a network. The attacker can only transmit data packets over the network, the attacker attempts to gain access to the machine, by exploiting some vulnerability.

Probing Attack (Prob): It is when an attacker attempts to acquire information from a network, for evading the systems, security protocols.

Since 1999, a large number of researchers assessed their IDS models using KDD Cup 99. This shows how KDD Cup 99 has been a working benchmark data set for over 15 years, and is still easily accessible and available today. The objective of the KDD 99 IDS competition is to create a standard data set for the surveying and evaluation of research in intrusion detection, [15]. Researchers found some difficulties or hurdles in training with KDD99, Olusola *et al.* [16] have analyzed the KDD 99 data set for selecting a relevant feature. They proposed that some features or attributes were not related to any attack, [17] they have taken 10% of the whole data set to perform their analysis.

IV. FORMULATING THE PROBLEM

Intrusion detection based on ANN is built by using gathered features about several types of attacks. Usually, building knowledge based on gathered data required sufficient amount of data with comprehensive nature. Unfortunately, in the application of intrusion detection, it is not feasible to create a sufficient knowledge for learning or at least balanced learning between the different classes (refer to the problem described in KDD99 in the previous section). Therefore, learning algorithm has to be carefully optimized according to the nature of the dataset. This leads us to investigate about how to identify the optimization parameters of the learning algorithm. In this work, the problem will be formulated as an optimization problem. More specifically, the problem is how to find the optimal values of the hidden layer neurons in both SLFN, and FLN in order to maintain highest accuracy of testing. Such problem is addressed in the literature as a heuristic searching in the space of solutions considering the aim is to minimize an objective function represents the accuracy of the classification of attacks. Mathematically, assuming that the accuracy of the testing is the function $\boldsymbol{f}(x)$, where $\boldsymbol{x} = (\boldsymbol{x_1}, \boldsymbol{x_2}, \ldots \boldsymbol{x_n})$ denotes the random selected different weights of hidden layer network. Our problem is presented in equation (1)

$$x^* = argmax\, f \quad \textbf{s.t.}\ (x_1, x_2, \ldots x_n) \in [-1, 1]^n \tag{1}$$

V. DEVELOPED METHODOLOGY

This section presents the developed methodology for this research. Firstly, particle swarm optimization is PSO presented in section. Secondly, particle Fast Learning Network (FLN) presented in section. Thirdly, our adaption of PSO to build FLN based training for IDS is presented in section.

A. PARTICLE SWARM OPTIMIZATION

Particle Swarm Optimization (PSO) is a parallel evolutionary computation technique developed by Mishra and Sengupta [23]. The protocol has been developed based on the social behavior metaphor. The PSO algorithm's performance is greatly influenced by the included tuning parameters, often referred to as the exploration– exploitation tradeoff: whereby exploration describes the ability to assess various regions in the problem space to an attempt to pinpoint a good optimum, preferably the global one. Exploitation describes the ability to focus the search within near vicinity of a promising candidate solution, to effectively and quickly locate the optimum.

Despite recent research efforts, the selection of the algorithm parameters remains empirical to a large extent [18].

The objective function of PSO algorithm used to evaluate its solutions, and operates upon the resultant fitness values.

Each particle saves its position, composed of the candidate solution and its evaluates fitness, and its velocity [19]. PSO algorithm has been used in many applications to solve many problems [20]–[24].

The modifications of the position and velocity are a process for seeking optimal solution at each iteration using as following:

$$\begin{aligned} v_i(k+1) &= wv_ik + c1r1(xbest, local - x_i) \\ &\quad + c2r2(xbest, global - x_i) \qquad (2) \\ x_i(k+1) &= xk + v(k+1) \qquad (3) \end{aligned}$$

The velocity and position of each particle are represented as the vectors $v_k = (v_{k1}, \ldots, v_{kd})$ and $x_i = (x_{i1}, \ldots, x_{id})$ respectively. In (2) x vectors are representing the best local and best global positions. $c1$ and $c2$ are acceleration factors known as cognitive and social parameters. $r1$ and $r2$ are random number between 0 and 1. k is the iteration index. w is the inertia weight parameter [25]. And update x_i for particle using (3).

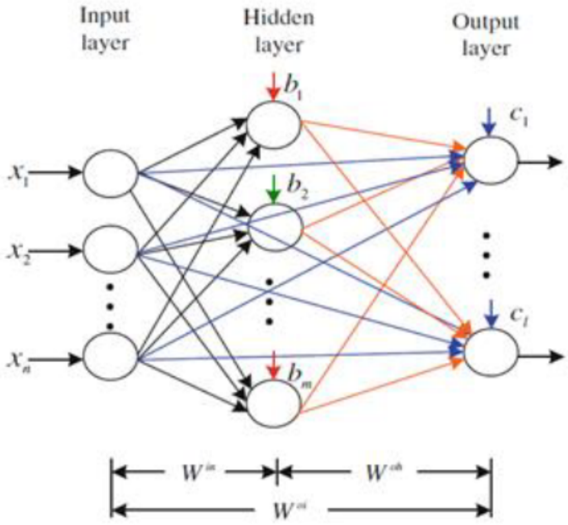

FIGURE 1. Structure of the FLN (Li et al., 2014).

B. FAST LEARNING NETWORK

The Fast Learning Network (FLN), proposed by Sahu *et al.* [15], is a parallel connection of an SLFN and a 3 layer FNN: input, hidden and output layer. FLN, an Artificial Neural Network, which is a Double Parallel Forward Neural Network (DPFNN), is demonstrated below using an analytical approach, namely the least square's methods as shown in Fig. 1

The FLN is essentially a DPFNN [16]. This describes a parallel connection of a multilayer FNN and a single-layer FNN. As discussed earlier, the re-coded external information from the hidden nodes, along with the external information itself directly from the input nodes is fed into the output nodes of the DFNNs. The FLN is mathematically modeled as [15]:

$$\begin{aligned} Y &= f\left(w^{io}x + w^{oh}G + c\right) = f\left(\left[w^{io}w^{oh}c\right]\begin{bmatrix} X \\ G \\ I \end{bmatrix}\right) \\ &= f\begin{pmatrix} & X \\ W & G \\ & I \end{pmatrix} \qquad (4) \end{aligned}$$

$$\begin{aligned} &\boldsymbol{G}(w^{in}, \ldots, w^{in}_m, b1 \ldots, bm, X_1, \ldots, X_N) \\ &\quad = \begin{bmatrix} g\left(w^{in}_1 x_1 + b_1\right) & \cdots & g\left(w^{in}_1 x_N + b_1\right) \\ \vdots & \ddots & \vdots \\ g\left(w^{in}_m x_1 + b_m\right) & \ldots & g\left(w^{in}_m x_N + b_m\right) \end{bmatrix}_{m\times N} \qquad (5) \end{aligned}$$

$$W = \left[w^{io}w^{oh}c\right]_{i\times(n+m+1)} \qquad (6)$$

$$I = [11 \ldots \ldots \ldots 1]\mathbf{1} \times N \qquad (7)$$

Where N represents the number of distinct samples in $\{x_i, y_i\}$, in which $x_i = [x_{i_1}, x_{i_2}, \ldots, x_{i_N}]^T \in R^n$ is the n-dimensional feather vector of the ith sample, and $y_i = [y_{i1}, y_{i2}, \ldots, y_{il}]^T \in R^l$ is the corresponding l - dimensional output vector. m represents the number of hidden layer nodes. w^{in} is the m × n input weight matrix, b = $(b_1, b_2, \ldots \ldots, b_m)$ represents the biases of the hidden layer nodes, and w^{oh} is a $\boldsymbol{l} \times \boldsymbol{m}$ matrix which consists of the weight values of the linking between the output layer and the input layer, C = $[c_1, c_2, \ldots \ldots, c_i]^T$ is the biases of output layer nodes. g(.) and f(.) represent the active functions of the hidden nodes and output nodes respectively., $w^{oi} = [w^{oi}_1, w^{oi}_2, \ldots \ldots \ldots . w^{oi}_i]$ represents the weight vector linking the jth output node and the input nodes, $w^{ok}_k = [w^{oh}_{1k}, w^{oh}_{2k}, \ldots \ldots \ldots . w^{oh}_{ik}]^T$ is the weight vector linking the kth hidden node and the output nodes, and $w^{in}_k = [w^{in}_{k1}, w^{in}_{k2}, \ldots \ldots \ldots, w^{in}_{km}]^T$ is the weight vector linking the kth hidden node and the input nodes. The matrix W = $[\mathbf{W}_{oi}\mathbf{W}_{oh}\mathbf{c}]$ could be called as output weights. G is considered the hidden layer output matrix of FLN.

C. PSO BASED OPTIMIZED FLN

As stated in the problem statement, FLN is similar to ELM in terms of lacking optimal weights, distribution or assignment. As a result, the overall accuracy of the ANN will be degraded unless a proper way in order to select the weights is performed. Our PSO-Based optimized FLN is trained based on selecting weights using particle swarm optimization.

PSO-based optimization of FLN is based on designing a particle that represents one candidate solution of FLN weights. One specific problem in performing the optimization is requiring to select both the weight's values as well as the number of neurons that are needed in the hidden layer of accomplish better accuracy. This means a variable length through the solution according to the number of the hidden neurons in FLN, and to overcome this problem, the maximum number of neurons in considered in assigning a length for the particle. For activation function, tanging has been used for the

TABLE 1. The testing accuracy of the benchmarks of comparison and the corresponding testing accuracy of PSO-FLN.

Number of neurons in the hidden layer	25	50	75	100	125	150	175	200
Basic ELM	0.9805	0.9847	0.9878	0.9891	0.99	0.9916	0.9923	0.9942
Online Sequential ELM	0.9806	0.9858	0.9883	0.9899	0.9903	0.9918	0.9931	0.9952
ATLBO-Based ELM	0.9817	0.986	0.988	0.9892	0.9905	0.9909	0.9924	0.9947
GA-Based ELM	0.9834	0.9873	0.988	0.9894	0.9909	0.9909	0.9924	0.9947
PSO-Based ELM	0.9815	0.9867	0.9859	0.99	0.9904	0.9921	0.9923	0.9951
HSO-Based ELM	0.9794	0.9865	0.9879	0.9889	0.9895	0.991	0.9922	0.9951
Basic FLN	0.9868	0.9889	0.9901	0.9912	0.9923	0.9932	0.9956	0.9957
GA-Based FLN	0.9885	0.9899	0.9904	0.9917	0.992	0.9926	0.9956	0.9969
ATLBO-Based FLN	0.9867	0.9885	0.9893	0.9905	0.9916	0.9939	0.9941	0.9949
PSO-Based FLN	0.9892	0.9895	0.9904	0.9907	0.9919	0.9927	0.9935	0.9968

output of the hidden layer neurons.

$$y = f(x) = \frac{2}{1 + e^{-2x}} - 1 \text{ where } x \in [-1, 1]$$

By using this function, we can cover the case of canceling out the neurons of the hidden layer network when the weights are selected to be zeros. In the following, the pseudo-code of PSO-based optimization is shown.

```
1. Create Initial Generation of Particles. $P_i = \{w_j\}$, $i = 1, \ldots N$ $J = 1, \ldots M$, $N$ population size, M weights number
2. For each particle do the following
3. For each particle build an equivalent FLN network.
4. For each FLN do the following
5. 1) Calculate accuracy of the FLN
6. 2) If the fitness value is better than the best local fitness value (pLBest) in history
7. Set current value as the new pLBest
8. End If
9. 3) If the fitness value is better than the best global fitness value (pLBest) set current
10. value as the new pLBest
11. Update particle position according to the position equation
12. Go to 4
```

VI. RESULTS AND DISCUSSION

In order to validate the developed learning model PSO-FLN, heavy comparison has been performed with a different number of neurons in the hidden layer of FLN, and the original ELM. Optimize the FLN parameters to enhance the IDS accuracy in our work, were proposed several algorithms such as Genetic algorithm (GA), Harmony Search Optimization (HSO) and Ameliorated Teaching Learning based optimization (ATLBO)26]. Also, these same optimization algorithms adoptive based ELM to compare with PSO-FLN as shown in table.1. And in Fig.2 we can see the relationship of accuracy with a number of neurons based on the results of Table 1, which presented the numbers as graphic. Results showed that PSO-FLN nearly outperformed other learning models regardless of changing the number of neurons in the hidden layer.

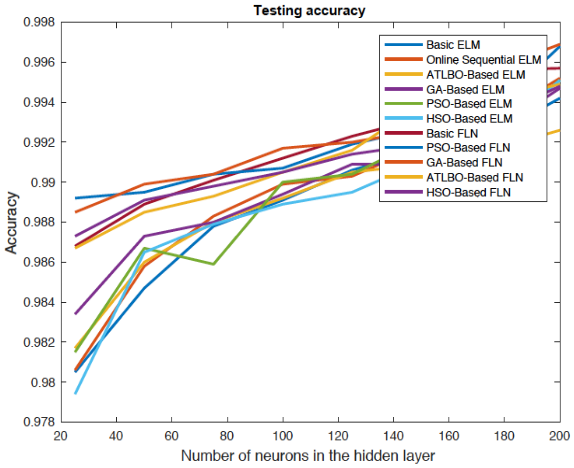

FIGURE 2. The testing accuracy of different optimization approaches with respect to the number of neurons.

TABLE 2. The confusion matrices: a. HSO based FLN, b. ATLBO based FLN, c. GA based FLN, d. PSO based FLN.

a.

	Normal	DOS	U2R	R2L	Prob
Normal	0.9970	0.0009	0.0003	0.0000	0.0011
DOS	0.0162	0.9811	0.0000	0.0000	0.0001
U2R	0.0676	0.0010	0.8040	0.0000	0.0019
R2L	0.2273	0.0000	0.0000	0.5455	0.1364
Prob	0.0943	0.0000	0.0000	0.0020	0.8001

b.

	Normal	DOS	U2R	R2L	Prob
Normal	0.9977	0.0009	0.0003	0.0000	0.0011
DOS	0.0200	0.9796	0.0004	0.0000	0.0001
U2R	0.1427	0.0152	0.8411	0.0000	0.0019
R2L	0.3182	0.0000	0.0000	0.5909	0.0909
Prob	0.1139	0.0079	0.0059	0.0020	0.8723

c.

	Normal	DOS	U2R	R2L	Prob
Normal	0.9974	0.0008	0.0003	0.0000	0.0013
DOS	0.0193	0.9799	0.0008	0.0000	0.0000
U2R	0.1446	0.0067	0.8478	0.0000	0.0010
R2L	0.2727	0.0000	0.0000	0.6064	0.1364
Prob	0.09 03	0.0000	0.0000	0.0020	0.8998

d.

	Normal	DOS	U2R	R2L	Prob
Normal	0.9978	0.0009	0.0003	0.0000	0.0011
DOS	0.0162	0.9837	0.0000	0.0000	0.0001
U2R	0.0676	0.0010	0.9363	0.0000	0.0010
R2L	0.2273	0.0000	0.0000	0.6364	0.0909
Prob	0.0943	0.0000	0.0000	0.0020	0.9077

Table.1 show that the increase of the hidden neuron's number linked to an increase the accuracy. For more details of the performance, the table (2) shows the confusion matrices of the testing results in all cases of optimize based on FLN with specified the hidden number of neurons 25. Our analysis with the confusion matrix results can be stated as follows. Class number 4, which represents R2L attack, has obtained lower accuracy comparing with other classes. This is due to the limited amount of training data in this class comparing with other classes. However, there is an obvious trend of improving accuracy with the increase to the number of neurons in the hidden layer.

The optimization algorithms were used based on the default situation of it. Firstly, we used HSO parameters set as the Harmony Memory (HM) $= 50$, Harmony Memory Considering Rate (HMCR)$= 0.7$ and Pitch Adjusting Rate (PAR)$= 0.35$. Secondly, for ATLBO, the number of learners $= 50$. Thirdly, the ga function from GA toolbox as in [27]. Fourthly, for PSO parameters formulae such as $c1 = c2 = 1.42$, $w = 0.75$ and number of particles $= 50$. As for parameters that share for all the optimization models, number of iterations $= 100$, training data $= 72788 \times 40$ and testing data $= 72798 \times 40$.

VII. CONCLUSION AND SUMMARY

In this article, the problem of intrusion detection has been presented and different approaches of solving were discussed. Using ANN based intrusion detection is more promising for reducing the number of wrong negative or false positives because ANN has the capability of learning from actual examples. A developed learning model for FLN based on particle swarm optimization has been proposed and named as PSO-FLN. The model has been applied to the problem of intrusion detection and validated based on the famous dataset KDD99. Our developed model has been compared against the wide range of meta-heuristic algorithms for training ELM, and FLN classifier. It can be concluded that our developed PSO-FLN has outperformed other learning approaches in the testing accuracy of the learning. Another finding is that the accuracy has increased for all models with increasing the number of hidden neurons in the ANN Future work is to counter the problem of less accuracy for a certain number of class because of the limited available amount of training data for such class.

REFERENCES

[1] B. G. Goodarzi, H. Jazayeri, and S. Fateri, "Intrusion detection system in computer network using hybrid algorithms(SVM and ABC)," *J. Adv. Comput. Res.*, vol. 5, no. 4, pp. 43–52, 2014.

[2] S. K. Gautam and H. Om, "Computational neural network regression model for host based intrusion detection system," *Perspectives Sci.*, vol. 8, pp. 93–95, Sep. 2016.

[3] F. A. Anifowose and S. I. Eludiora, "Application of artificial intelligence in network intrusion detection," *World Appl. Programm.*, vol. 2, no. 3, pp. 158–166, 2012.

[4] D. E. Denning, "An intrusion-detection model," in *Proc. IEEE Symp. Secur. Priv.*, vol. 2. Apr. 2012, pp. 118–131.

[5] M. S. Hoque, M. A. Mukit, M. A. N. Bikas, and M. S. Hoque, "An implementation of intrusion detection system using genetic algorithm," *Int. J. Netw. Secur. Appl.*, vol. 4, no. 2, pp. 109–120, 2012.

[6] J. Frank, "Artificial intelligence and intrusion detection: Current and future directions," in *Proc. 17th Nat. Comput. Secur. Conf.*, Baltimore, MD, USA, Oct. 1994. [Online]. Available: https://scholar.google.com/scholar?hl=en&as_sdt=0%2C5&q=A.+D.+O.+D.+Dabt–c-%2C+"Arti+cial+Intelligence+and+Intrusion+Detection?%3A+Current+and+Future+Directions+1+Problems+in+Intrusion+Detection+2+Arti+cial+Intelligence+Methods%2C"+Expert+Syst.%2C+pp.+1-12%2C+1994&btnG=

[7] M. Green, U. Ekelund, L. Edenbrandt, J. Björk, J. L. Forberg, and M. Ohlsson, "Exploring new possibilities for case-based explanation of artificial neural network ensembles," *Neural Netw.*, vol. 22, no. 1, pp. 75–81, 2009.

[8] S. Kiranyaz, T. Ince, A. Yildirim, and M. Gabbouj, "Evolutionary artificial neural networks by multi-dimensional particle swarm optimization," *Neural Netw.*, vol. 22, no. 10, pp. 1448–1462, 2009.

[9] T. W. S. Chow, J. Y. F. Yam, and S. Y. Cho, "Fast training algorithm for feedforward neural networks: Application to crowd estimation at underground stations," *Artif. Intell. Eng.*, vol. 13, no. 3, pp. 301–307, 1999.

[10] G.-B. Huang, X. Ding, and H. Zhou, "Optimization method based extreme learning machine for classification," *Neurocomputing*, vol. 74, pp. 155–163, Dec. 2010.

[11] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, "Extreme learning machine: Theory and applications," *Neurocomputing*, vol. 70, nos. 1–3, pp. 489–501, 2006.

[12] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, "Extreme learning machine: A new learning scheme of feedforward neural networks," in *Proc. IEEE Int. Joint Conf. Neural Netw.*, vol. 2. Jul. 2004, pp. 985–990.

[13] G. Li, P. Niu, X. Duan, and X. Zhang, "Fast learning network: A novel artificial neural network with a fast learning speed," *Neural Comput. Appl.*, vol. 24, nos. 7–8, pp. 1683–1695, 2014.

[14] J. Wang, W. Wu, Z. Li, and L. Li, "Convergence of gradient method for double parallel feedforward neural network," *Int. J. Numer. Anal. Model.*, vol. 8, no. 3, pp. 484–495, 2011.

[15] S. K. Sahu, S. Sarangi, and S. K. Jena, "A detail analysis on intrusion detection datasets," in *Proc. IEEE Int. Adv. Comput. Conf. (IACC)*, Feb. 2014, pp. 1348–1353.

[16] A. A. Olusola, A. S. Oladele, and D. O. Abosede, "Analysis of KDD'99 intrusion detection dataset for selection of relevance features," in *Proc. World Congr. Eng. Comput. Sci.*, Jan. 2016, pp. 16–23.

[17] T.-S. Chou, J. Fan, S. Fan, and K. Makki, "Ensemble of machine learning algorithms for intrusion detection," in *Proc. IEEE Int. Conf. Syst. Man Cybern.*, Oct. 2009, pp. 3976–3980.

[18] I. C. Trelea, "The particle swarm optimization algorithm: Convergence analysis and parameter selection," *Inf. Process. Lett.*, vol. 85, no. 6, pp. 317–325, 2003.

[19] J. Blondin. *Particle Swarm Optimization: A Tutorial*. [Online]. Available: http://cs.armstrong.edu/saad/csci8100/pso_tutorial.pdf

[20] A. Sengupta, S. Bhadauria, and S. P. Mohanty, "TL-HLS: Methodology for low cost hardware trojan security aware scheduling with optimal loop unrolling factor during high level synthesis," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 36, no. 4, pp. 660–673, Apr. 2017.

[21] V. K. Mishra and A. Sengupta, "Swarm-inspired exploration of architecture and unrolling factors for nested-loop-based application in architectural synthesis," *Electron. Lett.*, vol. 51, no. 2, pp. 157–159, 2015.

[22] A. Sengupta and S. Bhadauria, "User power-delay budget driven PSO based design space exploration of optimal k-cycle transient fault secured datapath during high level synthesis," in *Proc.-Int. Symp. Qual. Electron. Des. (ISQED)*, Apr. 2015, pp. 289–292.

[23] V. K. Mishra and A. Sengupta, "MO-PSE: Adaptive multi-objective particle swarm optimization based design space exploration in architectural synthesis for application specific processor design," *Adv. Eng. Softw.*, vol. 67, pp. 111–124, Jan. 2014.

[24] A. Sengupta and V. K. Mishra, "Expert systems with applications automated exploration of datapath and unrolling factor during power—Performance tradeoff in architectural synthesis using multi-dimensional PSO algorithm," *Expert Syst. Appl.*, vol. 41, no. 10, pp. 4691–4703, 2014.

[25] Y. Shi and R. Eberhart, "A modified particle swarm optimizer," in *Proc. IEEE Int. Conf.*, May 1998, pp. 69–73.

[26] G. Li, P. Niu, W. Zhang, and Y. Liu, "Model NOx emissions by least squares support vector machine with tuning based on ameliorated teachingŰlearning-based optimization," *Chemometrics Intell. Lab. Syst.*, vol. 126, pp. 11–20, Jul. 2013.

[27] J. Tvrdık, "Competitive differential evolution and genetic algorithm in GA-DS toolbox," *Tech. Comput. Prague, Praha, Humusoft*, vol. 1, no. 2, pp. 99–106, 2006.

MOHAMMED HASAN ALI received the M.Sc. degree in computer science from Universiti Teknikal Malaysia Melaka in 2014. He is currently pursuing the Ph.D. degree with Universiti Malaysia Pahang. He is currently a Research Assistant with Universiti Malaysia Pahang. He has authored over 10 research peer review articles in different journals and international conferences. His current research interests include Android security and network security.

BAHAA ABBAS DAWOOD AL MOHAMMED was born ThiQar, Iraq, in 1989. He received the B.Sc. degree in engineering from the Electromechanical System Department, University of Technology Iraq, in 2011. He is currently pursuing the master's degree with the Communications and Network Department, Wireless Communication Engineering Branch, Universiti Putra Malaysia, Malaysia. He is with the Department of Computer and Communication Systems, Faculty of Engineering, Universiti Putra Malaysia.

ALYANI ISMAIL (M'17) received the B.Eng. degree (Hons.) in electronic and information engineering from the University of Huddersfield, U.K., in 2000, and the M.Sc. degree in communication and computer and human-centered systems engineering (major in communication) and the Ph.D. degree in electronics engineering (thesis: design if microwave waveguides and filters for micromachining) from the University of Birmingham, U.K., in 2002 and 2006, respectively. She is currently a Lecturer with the Department of Computer and Communication Systems Engineering, Faculty of Engineering, Universiti Putra Malaysia, Malaysia. She is a member of the International Association of Engineers.

MOHAMAD FADLI ZOLKIPLI received the doctorate degree in computer science from Universiti Sains Malaysia in 2012. He is currently a Senior Lecturer with the Faculty of Computer Systems and Software Engineering, Universiti Malaysia Pahang. His career in academia started when he joined KUKTEM in 2002 as an academician. His teaching expertise includes data communication and networking, switching and routing, and network security. He is currently active in supervising research students of master and doctorate degrees. He has authored numerous articles in the area of computer systems and networking, especially in security domain such as intrusion detection systems, malware analysis, and cloud security. His research interests cover the broad area of digital security. As a part of the research community, he was also a reviewer for conferences and journals.