Signal Processing and Renewable Energy
December 2020, (pp. 81-94)
ISSN: 2588-7327
eISSN: 2588-7335

Detection of Attacks and Anomalies in The Internet of Things System Using Neural Networks Based on Training with PSO and TLBO Algorithms

Mohammad Nazarpour[1], Navid Nezafati[2*], Sajjad Shokouhyar[2]

[1] Department of Information Technology Management, Central Tehran Branch, Islamic Azad University, Tehran, Iran.
[2] Department of Management, Shahid Beheshti University, Tehran, Iran.

Received: 27-Sep-2020, Revised: 20-Oct-2020, Accepted: 21-Oct-2020.

Abstract

Detecting attacks and anomalies is one of the new challenges in commercializing and advancing IOT technology. One of the most effective methods for detecting attacks is the machine learning algorithms. Until now, many ML models have been suggested to detect attacks and anomalies, all of them use experimental data to model the detection process. One of the most popular and efficient ML algorithms is the artificial neural network. Neural networks also have different classical learning methods. But all of these classic learning methods are problematic for systems that have a lot of local optimized points or have a very complex target function so that they get stuck in local optimal points and are unable to find the global optimal point. The use of evolutionary optimization algorithms for neural network training can be an effective and interesting method. These algorithms have the capability to solve very complex problems with multi-purposed functions and high constraints. Among the evolutionary algorithms, the particle swarm optimization algorithm is fast and popular. Hence, in this article, we use this algorithm to train the neural network to detect attacks and anomalies of the Internet of Things system. Although the PSO algorithm has so many merits, in some cases it may reduce population diversity, resulting in premature convergence. So, in order to solve this problem, we make use of the TLBO algorithm and also, we show that in some cases, up to 90% accuracy of attack detection can be obtained.

Keywords: Attack detection, Neural Network, PSO Algorithm, Fuzzy rule, Adaptive Formulation, TLBO Algorithm.

***Corresponding Authors Email:** n_nezafati@sbu.ac.ir

1. INTRODUCTION

Today, the Internet of Things technology is growing fast and is promising for life in the smart world. The Internet of Things is trying to get data by connecting things from home appliances and medicine to high industrial equipment, transportation systems and meteorological equipment connected to the network (these data are the output of all kinds of sensors such as temperature, air pollution, smoke sensors, humidity sensors, RFID sensors, ultrasonic sensors, etc.). And after analysis and processing, it can make the appropriate responses output. This response can be a command to change the status of a device (for example, turning on the home air conditioning system) or a warning message to the user in the form of mobile software (air pollution warning). Increasing the scope of Internet of Things use increases energy consumption, makes management more complex, increases data volume, and needs high bandwidth to send data and high-speed processing systems. There are challenges in commercializing the Internet of Things technology that need to be addressed. One of the most important challenges is privacy and information security [1-3]. Admittedly, the satisfaction of consumers with this technology is related to these challenges. Today, IoT technology uses advanced technologies such as data transmission via fiber optic routes, the use of SDN (Software Defined Network) software's, the use of cloud computing and other up-to-date technologies to send and receive signals, processing and storage which increases the threat of attackers to infrastructure in this area. Threats and attacks that occur in the area of Internet of Things can be classified into four general categories: 1- Denial of service (DOS) attacks 2- Attacks of remote password detection risks (R2L) 3- Attacks of the dangers of discovering the user's password to the root (U2R) 4- searching and researching attacks (PROBING). One way to deal with attacks and threats on the Internet of Things is to use machine learning models and mechanisms [4-5]. Machine learning is a subset of Artificial Intelligence technology that is mainly based on machine learning based on the machine's own experiences and predictions that emerged from those experiences. Machine learning algorithms, using a set of data called training data set and create the required models. When new data is introduced into the machine learning algorithm, the system can perform the predictive process based on the model created. One of the most famous widely used and strongest algorithms and models for learning machines is the Artificial Neural Network [6-7]. The purpose of this paper is to identify the threats and attacks of an Internet of Things system through a modified artificial neural network. The aim of this paper is to identify the threats and attacks of an Internet of Things system by a modified artificial neural network. Neural networks are highly organized network structures that are modeled on the functioning of the human nervous system. The neural networks of the three layers of input, middle, and output are formed and connected by neurons. In neural networks, information is received through input neurons. The middle layers and neurons, which can be multi-layered, receive this information, then process and analyze it. This transfer of information continues until they reach the output layer. The neural

network also uses a mathematical or computational model to process information, which acts on the basis of the connector-to-computational approach. One of the classic types of the artificial neural network is the Perceptron network. Figure 1 shows a schematic of this network and biological neural network.

In the artificial neural network, the X1 to Xn inputs enter the input neurons. Each of these inputs is then multiplied by the coefficients W. Notice that the training process is on the basis of this input data and the W coefficients are diagnosed at the end of the process. The product of multiplication of fixed inputs (w) at X inputs is added to the constant values called bias (b), which must also be diagnosed in the training process. Lastly, all of these values are entered into the output layer. The last layer encompasses an activation function, which is a nonlinear function. So far, various methods have been proposed for training neural networks, or in other words, determining the coefficients w for a data set of a system. The most common method of training is the BP return method because this system has the desired performance speed in finding the optimal response of W coefficients [8-9]. In this method, there are two steps in each iteration. In the first step, the initial W coefficients are multiplied by the input, and the operation continues until we reach the output, and this output is possibly very far from the actual output. Then, the error between the actual output and the output calculated by the neural network is calculated. Now that we understood how much error the algorithm has in terms of weights and deviations, we move on to the second step in one iteration. At this point, we can go back and synchronize the weights and deviations. That is, we change the weights and deviations so that in the next iteration they produce a result closer to the actual output with less error. Unfortunately, all of these algorithms, which act on an error slope, have difficulty such as getting stuck in the local minimum point and being unable to obtain a global response [10-11]. This getting stuck in the local minimum response leads to the neural network training flow to stop before reaching the original optimal response. With this training process, the neural network is not able to provide an accurate model of the system, and we need to look at more exact training methods. One possible way to get an accurate model of a neural network-based system is to use the evolutionary optimization algorithms. The evolutionary optimization algorithms are able to optimize and solve very complex problems as well as multi-purposed problems. One of the evolutionary optimization algorithms that have a good performance speed is the PSO algorithm. Additionally, this algorithm, like the other evolutionary algorithms (genetics and colonial competition and so on) has simpler calculations. In this article, we used the PSO algorithm to train the neural network. Then, we will show that although training by the PSO algorithm gives a much more accurate answer than the BP training method, it is still possible to reach much more accurate answers by changing the PSO algorithm. For this purpose, we used a combination of fuzzy, comparative and mutation methods to alter this algorithm and showed that we get very acceptable results by training the neural network by the altered PSO algorithm.

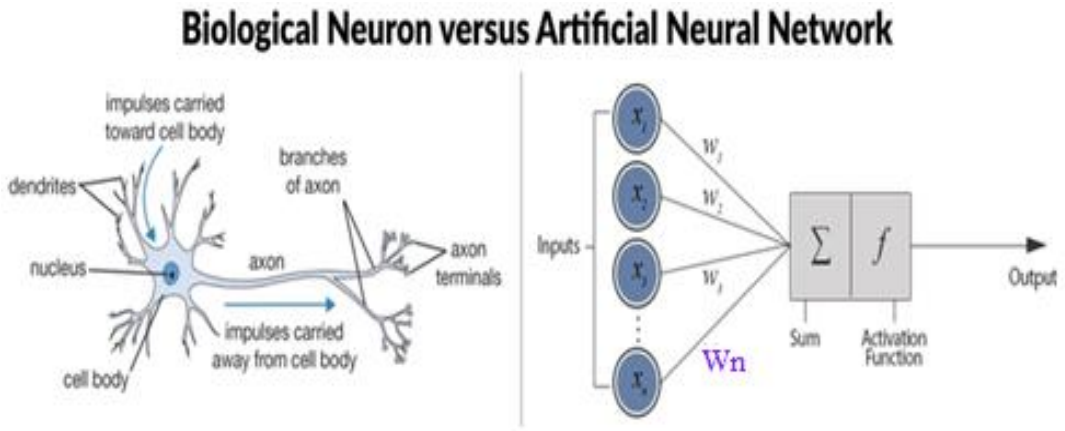

Fig. 1. Schematic of biological neuron (Left) versus artificial neural network (Right).

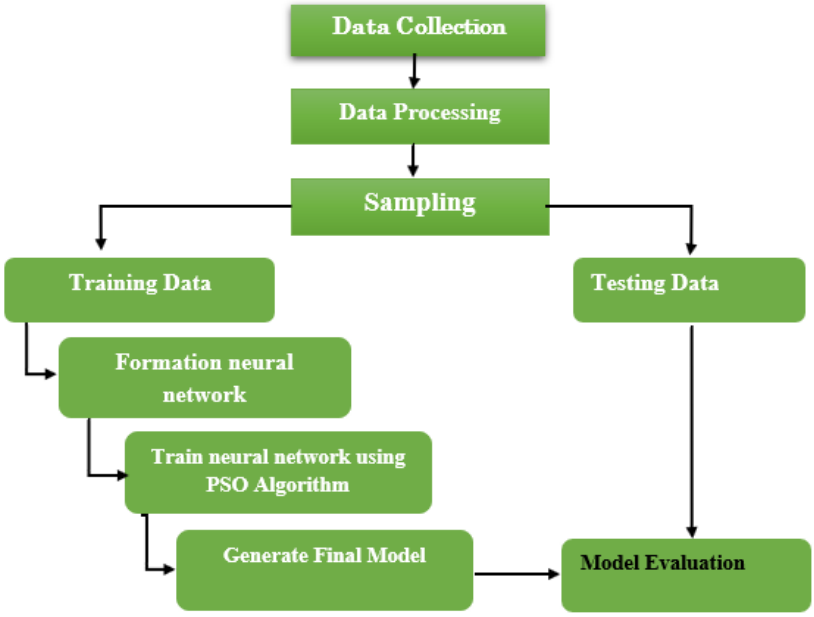

Fig. 2. Overall framework of the attack detection using neural-network based PSO algorithm.

2. MATERIALS AND METHODS SECTION

The general structure and flowchart of the proposed design are shown in Figure 2. As shown in the figure, in the first step the data of the proposed system should be collected. This article uses data from the KDD-CUP collection. The second step is to pre-process the data, including clearing the similar data, normalizing the data, engineering the data feature, and removing them. Then, the data which have been compacted are divided into two categories of training data (80% data) and testing data (20% data). The neural network now models the system from the training data using the data it sees through the

PSO algorithm and TLBO algorithm. Last, we use testing data to evaluate the model and calculate the modeling accuracy.

3. CLASSICAL PSO

Particle swarm optimization algorithms are among the evolutionary optimization algorithms. The most important advantage of these algorithms over the other optimization algorithms is that they do not require complex operations and mathematical relationships such as derivatives and integrals [12-13]. These algorithms are either modeled on the basis of the biological processes and exchanges of organisms (such as ants, birds, genetics, etc.) or human socio-political exchanges and behaviors (such as colonial competition algorithms or Teacher Learn Based Optimization) [14-15]. The PSO algorithm is also modeled based on the search for suitable habitat by birds. This algorithm was proposed and invented in 1995 by a joint study of Eberhart and Kennedy based on the movement of birds and fish on the basis of the two principles of artificial life and evolution. Like other evolutionary algorithms, this algorithm starts with a set of particles of a matrix with a complete random position. Each particle in this matrix is called a bird, and these particles can fly in the nth -aspect space (n is the number of variables in the optimization problem). And at each step, their new condition is updated on the basis of the past personal experiences and the situation of their neighbors. The strength of each particle of this set of birds is defined by the following vector [16-18]:

$$Xi = [X_{i1}, X_{i2}, \ldots\ldots X_{in}]^T \in S \quad (1)$$

In this regard, S is the search space and Xi is the position of each particle in repeating i algorithm. Each particle has a velocity at each step. Therefore, the velocity vector of all particles is defined by relationship 2 [16-18]:

$$Vi = [V_{i1}, V_{i2}, \ldots\ldots V_{in}]^T \in S \quad (2)$$

The best personal position that each particle has from the beginning to the **i** step is called the best personal position and is defined for all particles by the following vector in each step [16-18]:

$$Pi = [P_{i1}, P_{i2}, \ldots\ldots P_{in}]^T \in S \quad (3)$$

Based on the relationships and definitions described above, the rate and speed of each particle at each step of repetition is calculated and updated by the following relationship [16-18]:

$$\vec{v}_i^{\,k+1} = w\vec{v}_i^{\,k} + c_1 r_1 \times (\vec{p}_i - \vec{x}_i^{\,k}) + c_2 r_2 \times (\vec{p}_g - \vec{x}_i^{\,k}) \quad (4)$$

$$\vec{X}_i^{\,k+1} = \vec{V}_i^{\,k+1} + \vec{X}_i^{\,k} \quad (5)$$

In this regard, the updated speed of the particle is in the repetition of k + 1 and the previous velocity and position of the particle respectively. It is also the best I-th particle position ever as well as the position of a particle that has the p-best among particles. Here c1 and c2 are constant coefficients and are usually 2. If the value of c1 increases, the particle tends to follow the search around its best personal position. However, if c2 is greater than c1, the inclination of the particle is to search around the global position. Hence, it is better to reconcile the process of choice between these two parameters. The coefficient w is known as the inertial weight

coefficient. This coefficient determines the impact of the previous speed on the new speed. If the small w coefficient is selected, the search step is short and consequently, the search space is small and of course, the search accuracy increases. However, if selected number is large, the search step and the search space for each particle will be longer, but the search accuracy will be lower. r1 and r2 are two random numbers between zero and one that give a random nature to the search pattern. In many cases, the w coefficient is constant and about0.9. However, in some cases it is linear and a function of program repetition. So, first a large search is selected to enlarge the search space at the beginning of the search. Then, with increasing the iteration pattern, its value decreases so that the further we go, the more accurate result we obtain. Although this method gives a more accurate answer than the choice of w with a constant value, it still cannot be applicable in all engineering issues. Therefore, it is then selected by fuzzy rules in a comparative manner. If the target function is close to the optimal value, the coefficient w is small and if it is far away, the coefficient w is selected.

4. FUZZY RULES FOR DIAGNOSING THE INERTIAL COEFFICIENT W

The weight factor W has a huge impact on the speed of each particle at the current stage, so increasing this factor increases the speed. Since it is supposed that in relationship number 5, the amount of each displacement is considered one second, so the higher the speed, the higher the particle displacement in one step, and consequently, the search space is large and its accuracy decreases. The opposite is true. Hence, an appropriate balance must be taken into account in selecting this particle. In this article, this equilibrium is performed using fuzzy rules and ifs. The best choice is to match the w coefficient to whether Gbest is close to or far from each step of the desired Gbest using fuzzy logic. Here, the values of w and NFV, which are defined below, are the inputs of the fuzzy inference motor and its output is Δw [19-20].

$$NFV = \frac{(FV - FV_{\min})}{(FV_{\max} - FV_{\min})} \qquad (6)$$

Here FV is the Gbest level in the current step and FVmin is the Gbest level in the first repetition and FVmax is a very large number. Usually, the W coefficient must be between 0.9 and 0.4. Since the correction of the W factor during the implementation of the program may be increasing or decreasing, both positive and negative corrections are essential for this coefficient. In this research, a small number with a value of 0.1 is regarded, which is added and subtracted by the W factor.

$$\omega^{k+1} = \omega^{k} + \Delta\omega \qquad (7)$$

Here Δw is a similar correction value and is equal to ± 1. Of course, sometimes the value is zero and its status is suggested according to Table 1. Notice that Gbest values must be expressed as the membership functions to attain an optimal value for the weight factor W. In this article, it is recommended that triangular membership functions were selected so that they have three states:

Large or L, small or M, and medium or M, also, the fuzzy model outputs, as shown in

Table 1, have three values of PE ((+0.1, NE ((-0.1) or ZE (0). As shown in Table 1, the 9 states may occur based on different values of NFV and W. If both NFV and W are small, there is no need to change w because on the one hand, Gbest has reached the optimal level and on the other hand, it is not possible to decrease W so much that it excels the permutable limit. If the NFV is low and the W is medium, you can still reduce the W by 0.1 to increase the search accuracy. If the NFV is low and the W is high, you can reduce the w by 0.1 as much as before. Here the relationship between inputs and outputs is showed in Table 1. Also, the triangular membership functions are represented in Figure 3. These functions are used to get the input and output variables.

5. TLBO (TEACHER LEARN BASED OPTIMIZATION) ALGORITHM

In recent years, meta-heuristic algorithms have been used to optimize engineering problems. These algorithms are either modeled based on natural phenomena (such as ant colony and birds' algorithms) or sample human social exchanges (such as Imperial competition algorithms and teacher learning algorithm). The most important advantage of these algorithms is that they are simple and do not require complex mathematical problems such as derivative and integral. The Teacher Learn Based Optimization algorithm is an interesting algorithm for optimizing engineering issues where is modeled based on the teacher training in the classroom. This algorithm has two training steps. The first step is based on teacher training and the second one is based on student debate after the end of the class. In the first phase, the person who has the best answer in the population is selected as the teacher ($X_{teacher}$) and other members of the population are known as students (X_i). In the following, we calculate the average position of the students (X_{mean}). The reason for calculating the student knowledge average is that the teacher gives the training according to the average level of the class. By considering "r" as a random number as well as T_f as a constant coefficient, it is possible to model the movement of students in the first step by the following relation [21-24]:

$$\mathbf{X}_{new} = \mathbf{X}_i + r.(\mathbf{X}_{teacher} - T_f.\mathbf{X}_{mean}) \quad (8)$$

Here X_i and X_{new} are the current and the new situation of the students respectively, Tf is a training factor that is considered as 2.

Table 1. ***Fuzzy rules of the input and output variables.***

ΔW		*W*		
		S	*M*	*L*
NFV	*S*	*ZE*	*NE*	*NE*
	M	*PE*	*ZE*	*NE*
	L	*PE*	*ZE*	*NE*

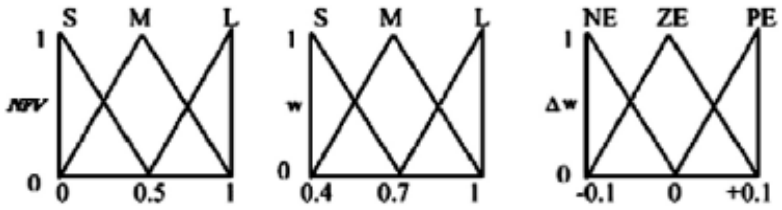

Fig. 3. The membership functions.

In the second stage, the teaching process is the responsibility of the students, so that each student selects another student randomly and shares knowledge with each other and also updates his / her position; thus, trying to use the other students' information to raise his / her level of awareness and knowledge. This phase can be modeled as following formulations [21-24]:

$$X_{new} = X_i + r(X_i - X_j) \ \ if \ \ f(X_i) < f(X_j) \quad (9)$$

$$X_{new} = X_i + r(X_j - X_i) \ \ if \ \ f(X_j) < f(X_i) \quad (10)$$

where is in this stage, the move is made if the new position is better than the previous position. Moreover, the condition for the termination of this algorithm is to reach the end of the iteration.

6. THE OBJECTIVE FUNCTION

In this research, to model the attack detection system and anomalies, we used the multilayer perceptron neural network structure as ML. Additionally, we trained the neural network using the BP algorithms, classical PSO algorithms, FPSO (fuzzy PSO), and TLBO algorithm. Moreover, we used the sigmoid function as the last layer of the neural network according to the following formula.

$$a(z) = \frac{1}{1 + \exp(-z)} \quad (11)$$

The accuracy of the suggested model is calculated on the basis of the correct detection of the model attained by the neural network and by the following relationship:

$$Accuracy = \frac{True\ Positive + \ True\ Negative}{True\ Positive + \ True\ Negative + False\ Positive \ + \ False\ Negative} \quad (12)$$

Since the PSO algorithm inherently minimizes the target function, the following function should be defined to increase the accuracy of the target function:

Cost Function = - accuracy (13)

7. RESULTS AND DISCUSSION SECTION

As referred to the previous section, the PSO algorithm is a powerful algorithm for finding optimal points in complex and multi-purpose problems. Hence, in this article, the neural network has one hidden layer with 15 neurons and training is done by the PSO algorithm. However, the classic model of this algorithm has a number of coefficients that if selected consistently decrease particle diversity and premature convergence, resulting in localized optimal locations. So, in this paper, the weighted coefficient of inertia is determined using rolls and the fuzzy logic rules. This operator is expected to curb the algorithm from getting trap in the optimal local locations. So, we used the TLBO algorithm and taught them the neural network and compared the outputs. Figure 4 shows the accuracy level for different neural network training methods. Here we suppose that the maximum repetition is equal to 50 and also the number of particles is equal to 40. As represented in the figure, the neural network training by the classical PSO algorithm is much more optimal than training by the BP

algorithm. Moreover, as expected, the classic PSO algorithm was entangled at the local optimal point, and the combination of FPSO gave the more accurate response. Also, the TLBO gives good results.

Figure 5 shows the convergence speed of different algorithms drowned on the iteration of the algorithm for diagnosing different attacks. As shown in the figure, the TLBO algorithm, in addition to being much more accurate, has a better convergence pace. So, this algorithm is a very optimal algorithm to increase the accuracy and speed of attack detection.

Table 2. ***Input parameters of Neural network.***

S/N	Name	Type	S/N	Name	Type
1	duration	Continuous	25	serror_rate	Continuous
2	protocol_type	Symbolic	26	srv_serror_rate	Continuous
3	service	Symbolic	27	rerror_rate	Continuous
4	flag	Symbolic	28	srv_rerror_rate	Continuous
5	src_bytes	Continuous	29	same_srv_rate	Continuous
6	dst_bytes	Continuous	30	diff_srv_rate	Continuous
7	land	Symbolic	31	srv_diff_host_rate	Continuous
8	wrong_fragment	Continuous	32	dst_host_count	Continuous
9	urgent	Continuous	33	dst_host_srv_count	Continuous
10	hot	Continuous	34	dst_host_same_srv_rate	Continuous
11	num_failed_logins	Continuous	35	dst_host_diff_srv_rate	Continuous
12	logged_in	Symbolic	36	dst_host_same_src_port_rate	Continuous
13	num_compromised	Continuous	37	dst_host_srv_diff_host_rate	Continuous
14	root_shell	Continuous	38	dst_host_serror_rate	Continuous
15	su_attempted	Continuous	39	dst_host_srv_serror_rate	Continuous
16	num_root	Continuous	40	dst_host_rerror_rate	Continuous
17	num_file_creations	Continuous	41	dst_host_srv_rerror_rate	Continuous
18	num_shells	Continuous			
19	num_access_files	Continuous			
20	num_outbound_cmds	Continuous			

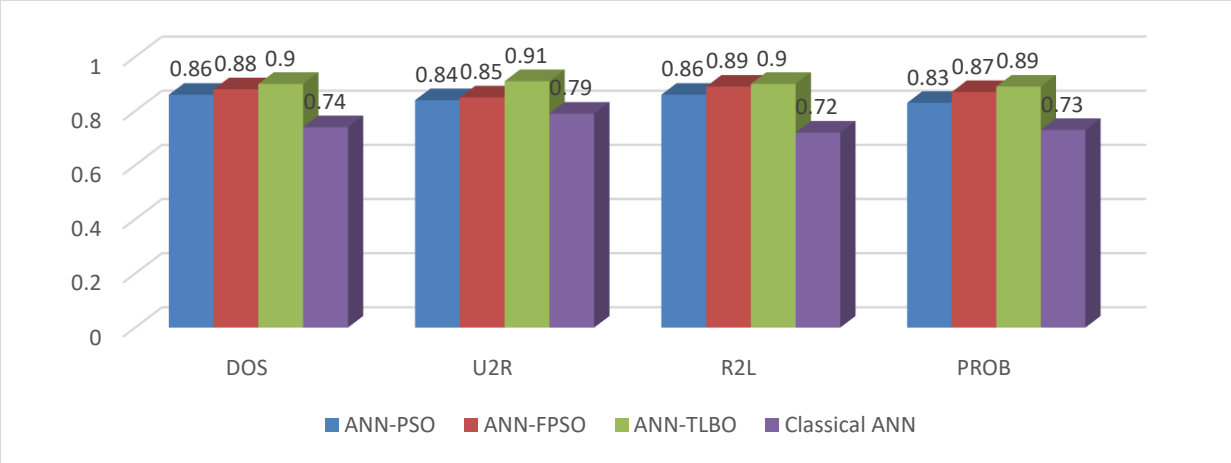

Fig. 4. Accuracy for different machine learning algorithm.

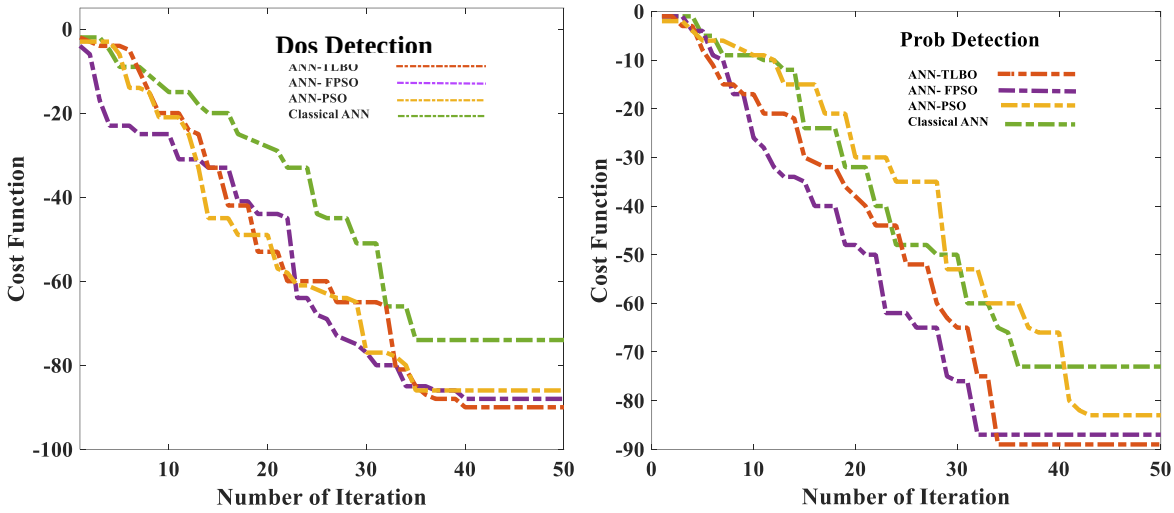

Fig. 5. The convergence characteristic of proposed method in different attack detection.

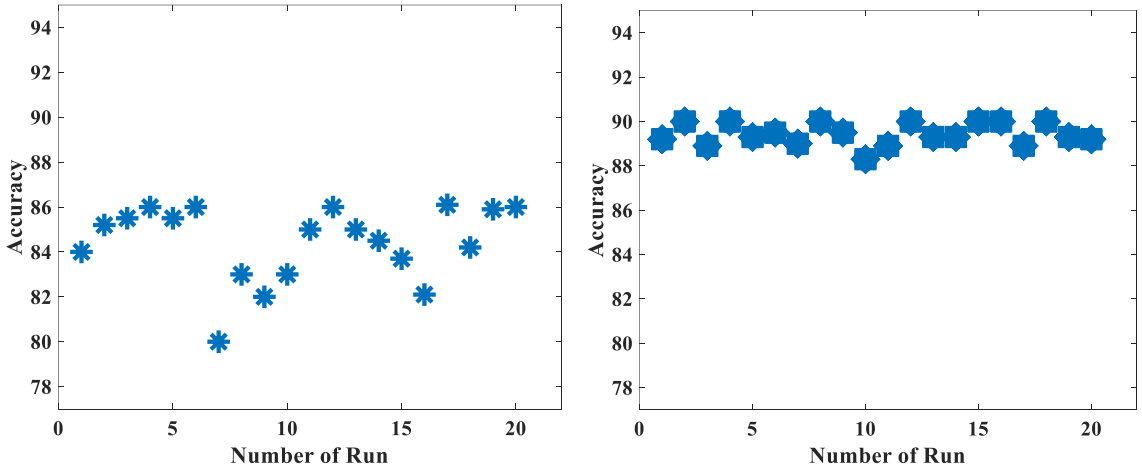

Fig. 6. Accuracy for 20 runs of the left algorithm: ANN-PSO right: ANN TLBO.

Table 3. ***Output Parameters of Neural Network (Attack Type).***

S/N	Name	Type
1	Back	dos
2	buffer_overflow	u2r
3	ftp_write	r2l
4	guess_passwd	r2l
5	imap	r2l
6	ipsweep	probe
7	land	dos
8	loadmodule	u2r
9	multihop	r2l
10	neptune	dos
11	nmap	probe
12	perl	u2r
13	phf	r2l
14	pod	dos
15	portsweep	probe
16	rootkit	u2r
17	satan	probe
18	smurf	dos
19	spy	r2l
20	teardrop	dos
21	warezclient	r2l
22	warezmaster	r2l

Lastly, in Figure 6, we show the accuracy of the PSO and TLBOO algorithms after running the program 20 times to detect Dos attacks. As shown in the figure, the TLBO algorithm is more dependable than the PSO algorithm. Due to different performances, the program relatively represents the same answers.

8. CONCLUSION

In this article, firstly we used an artificial neural network with the PSO optimization

algorithm to diagnose attacks and anomalies in the structure of the Internet of Things and showed that it could act better than traditional neural network training methods such as BP. Moreover, we showed that the PSO algorithm has a number of coefficients that are regarded constant in its classical type, leading to premature convergence of the algorithm and being getting stuck at the local minimum point. Therefore, we used comparative functions with fuzzy systems to diagnose the coefficients, and we were able to show that the accuracy and speed of the algorithm increased to some extent. But from the accuracy point of view, we were looking for a better training method. Using the TLBO algorithm, we came up with a very powerful neural network training algorithm to diagnose attacks and anomalies in the structure of the Internet of Things. The suggested ANN-TLBO algorithm is about 90% (90% for Dos type attack, 91% for U2R, 90% for R2L and 89% for PROB), and the accuracy for the PSO-ANN algorithm is about 86%.

REFERENCES

[1] Hasan, Mahmudul, et al. "Attack and anomaly detection in IoT sensors in IoT sites using machine learning approaches." Internet of Things 7 (2019): 100059.

[2] Kotenko, Igor, et al. "Attack detection in IoT critical infrastructures: a machine learning and big data processing approach." 2019 27th Euromicro International Conference on Parallel, Distributed and Network-Based Processing (PDP). IEEE, 2019.

[3] Foley, John, Naghmeh Moradpoor, and Henry Ochen. "Employing a Machine Learning Approach to Detect Combined Internet of Things Attacks against Two Objective Functions Using a Novel Dataset." Security and Communication Networks 2020 (2020).

[4] Doshi, Rohan, Noah Apthorpe, and Nick Feamster. "Machine learning ddos detection for consumer internet of things devices." 2018 IEEE Security and Privacy Workshops (SPW). IEEE, 2018.

[5] Syed, Naeem Firdous, et al. "Denial of service attack detection through machine learning for the IoT." Journal of Information and Telecommunication (2020): 1-22.

[6] Manimurugan, S., et al. "Effective Attack Detection in Internet of Medical Things Smart Environment Using a Deep Belief Neural Network." IEEE Access 8 (2020): 77396-77404.

[7] Latif, Shahid, et al. "A Novel Attack Detection Scheme for the Industrial Internet of Things Using a Lightweight Random Neural Network." IEEE Access 8 (2020): 89337-89350.

[8] Alkronz, Eyad Sameh, et al. "Prediction of Whether Mushroom is Edible or Poisonous Using Back-propagation Neural Network." (2019).

[9] Wang, Weilin, et al. "Estimation of PM2. 5 concentrations in China using a spatial back propagation neural network." Scientific reports 9.1 (2019): 1-10.

[10] Mohammadi, Farzaneh, et al. "Modelling and optimizing pyrene

removal from the soil by phytoremediation using response surface methodology, artificial neural networks, and genetic algorithm." Chemosphere 237 (2019): 124486.

[11] Azimi, Yousef, Seyed Hasan Khoshrou, and Morteza Osanloo. "Prediction of blast induced ground vibration (BIGV) of quarry mining using hybrid genetic algorithm optimized artificial neural network." Measurement 147 (2019): 106874.

[12] Cai, Jianghui, et al. "A Novel Clustering Algorithm Based on DPC and PSO." IEEE Access 8 (2020): 88200-88214.

[13] Singh, Shakti, Prachi Chauhan, and NirbhowJap Singh. "Capacity optimization of grid connected solar/fuel cell energy system using hybrid ABC-PSO algorithm." International Journal of Hydrogen Energy (2020).

[14] Devarasiddappa, D., M. Chandrasekaran, and R. Arunachalam. "Experimental investigation and parametric optimization for minimizing surface roughness during WEDM of Ti6Al4V alloy using modified TLBO algorithm." Journal of the Brazilian Society of Mechanical Sciences and Engineering 42.3 (2020): 1-18.

[15] Qiao, Weibiao, Hossein Moayedi, and Loke Kok Foong. "Nature-inspired hybrid techniques of IWO, DA, ES, GA, and ICA, validated through a k-fold validation process predicting monthly natural gas consumption." Energy and Buildings (2020): 110023.

[16] Prithi, S., and S. Sumathi. "LD2FA-PSO: A novel Learning Dynamic Deterministic Finite Automata with PSO algorithm for secured energy efficient routing in Wireless Sensor Network." Ad Hoc Networks 97 (2020): 102024.

[17] Kacimi, Mohand Akli, et al. "New mixed-coding PSO algorithm for a self-adaptive and automatic learning of Mamdani fuzzy rules." Engineering Applications of Artificial Intelligence 89 (2020): 103417.

[18] Jallal, Mohammed Ali, Samira Chabaa, and Abdelouhab Zeroual. "A novel deep neural network based on randomly occurring distributed delayed PSO algorithm for monitoring the energy produced by four dual-axis solar trackers." Renewable Energy 149 (2020): 1182-1196.

[19] Niknam, Taher, Ehsan Azadfarsani, and Masoud Jabbari. "A new hybrid evolutionary algorithm based on new fuzzy adaptive PSO and NM algorithms for distribution feeder reconfiguration." Energy Conversion and Management 54.1 (2012): 7-16.

[20] Niknam, Taher, Hassan Doagou Mojarrad, and Majid Nayeripour. "A new fuzzy adaptive particle swarm optimization for non-smooth economic dispatch." Energy 35.4 (2010): 1764-1778.

[21] Akhlaghi, Majid, Farzin Emami, and Najmeh Nozhat. "TLBO algorithm assisted for designing plasmonic nano particles based absorption coefficient." Optoelectronics and Advanced Materials-Rapid Communications

8.September-October 2014 (2014): 845-848.

[22] Akhlaghi, Majid, Farzin Emami, and Najmeh Nozhat. "Binary TLBO algorithm assisted for designing plasmonic nano bi-pyramids-based absorption coefficient." Journal of Modern Optics 61.13 (2014): 1092-1096.

[23] Akhlaghi, Majid. "Optimization of the plasmonic nano-rods-based absorption coefficient using TLBO algorithm." Optik 126.24 (2015): 5033-5037.

[24] Balvasi, Mohsen, Majid Akhlaghi, and Hossein Shahmirzaee. "Binary TLBO algorithm assisted to investigate the supper scattering plasmonic nano tubes." Superlattices and Microstructures 89 (2016): 26-33.