

<http://ansinet.com/itj>

ITJ

ISSN 1812-5638

INFORMATION TECHNOLOGY JOURNAL

ANSI*net*

Asian Network for Scientific Information
308 Lasani Town, Sargodha Road, Faisalabad - Pakistan

Network Intrusion Detection Based on IPSO-BPNN

¹Yuhong Zhao, ²Yirui Zhao and ³Xuecheng Zhao

^{1,2}School of Electric Engineering, University of south China, Hengyang, 421001, Hunan, China

³Mechanical and Electrical Engineering Department, Shaoyang Vocational and Technical,
Shaoyang, Hunan, China

Abstract: In order to resolve the problem of high false positive rate of traditional intrusion detection algorithm, the hybrid algorithm which combines Improved Particle Swarm Optimization (IPSO) with Back Propagation (BP) neural network algorithm is used in computer network intrusion detection in this study. Based on the characteristics of the local precise search of the BP networks and the global search of the improved particle swarm optimization algorithm, this method optimizes the weight of the BP networks, conquers the disadvantages of the BP networks that are easily trapped in the local extremum. Then the network structure is applied into intrusion detection systems. Comparing the simulation results of the KDD99 CUP data set with the intrusion detection system based on the traditional BP networks and the IPSO-BPNN algorithm, it can be seen that the training samples of the proposed intrusion detection algorithm have less time, with better recognition rate and detection rate.

Key words: Network intrusion detection, particle swarm optimization, neural network

INTRODUCTION

Due to the diversification, complex and intelligent of network attacks, relying on static defense such as the traditional operating system strengthening technology and firewall isolation techniques have been difficult to fit for the need of network security. Intrusion detection technology as the first step in the active defense, is one of indispensable technology to information security protection and become a new hotspot of network security technology in recent years. As the second defense line of the computer network security, network intrusion detection is to monitor the operational status of the network and find all kinds of attack attempts, aggressive behavior, or the result of the attack, in order to ensure the availability, integrity and confidentiality of the system resources (Mukherjee and Hebedein, 2004; Welch and Lathrop, 2003; Ye *et al.*, 2002).

With the exponential outspread and development of computer network technology, network intrusion presents a comprehensive and integrated evolution trend, which makes the currently used rule based misuse detection system inefficient, thus cause high false alarm rate and undetected rate. So, it is critical to develop novel methods for anomaly detection. Over the years, experts and scholars at home and abroad have carried on the extensive research to network intrusion detection. Such as intrusion detection system based on neural network, a distributed intrusion detection system based on agent,

intrusion detection based on support vector machine and intrusion detection based on data mining and some hybrid intrusion detection system, etc., the most significant features is the intrusion detection system developing in the direction of artificial intelligence.

The neural network has self-learning and adaptive ability. As long as the system audit data or network packets are provided, neural network can extract the characteristic mode from normal user or system activity and detect the attack mode from unusual activity by self-learning. BP neural network has the strongly nonlinear fitting ability and the learning rule is simple and easy computer, so BP learning algorithm is generally used for the training of neural network. However, the inherent defects such as easy to fall into local minimum point, slow convergence affect the prediction accuracy and therefore subject to certain restrictions in intrusion detection (Wang, 2010; Zhou and Wang, 2007; Clerc and Kennedy, 2002). Particle Swarm Optimization (PSO) algorithm is an evolutionary algorithm simulating birds and fish feeding, migration and aggregation behavior. The algorithm is simple, flexible and easy to realize. The algorithm preliminary search is fast but in the search of late, the speed is slower and slower, particle swarm shows strong convergence and easy to fall into local optimum. Against the premature convergence of PSO, in this study, the PSO algorithm is improved and the improved particle swarm optimization algorithm is used to train neural network parameters, so as to optimize the network, realize Back

Propagation (BP) neural network parameters automatic optimization (Van den Bergh, 2002). The results showed that the convergence speed of the proposed method is faster and the number of iterations is less, so the accuracy of the intrusion detection system can be improved to a certain extent.

PSO ALGORITHM INTRODUCTION

Particle Swarm Optimization (PSO) is an evolution computing technology based on swarm intelligence, which was proposed by Eberhart and Kennedy in 1995 and it is based on the simulation of the flock (Yi and Ge, 2005). The principle is described as follows: a flock of birds search food randomly in a region where there is only a piece of food. The most simple and effective strategy to find food is to search the area around the nearest bird from food, which constitutes one of the basic concepts of the particle swarm optimization. Assuming that a community is composed of n particles in a D -dimensional search space, where in $x_i = (x_{i1}, x_{i2}, \dots, x_{id})$, $i = 1, 2, \dots, n$, is the position of i -th particle in D -dimension. Fitness value can be calculated by substituting X_i into an objective function and the pros and cons of X_i can be obtained according to the size of the fitness value. "Flying" velocity of the particle i is a D -dimensional vector, denoted as $V_i = (V_{i1}, V_{i2}, \dots, V_{id})$, $P_i = (P_{i1}, P_{i2}, \dots, P_{id})$ is the best position of particle i searched, corresponding to the optimal solution particle i found by itself, which is the location of the best fitness value. Optimal position for the entire particle swarm search after the h -th iteration is $P_{igd} = (P_{i1}, P_{i2}, \dots, P_{id})$. By the above definition, the standard formula of particle swarm optimization can be expressed as:

$$V_{id}^{(k+1)} = \omega V_{id}^{(k)} + c_1 r_1^{(k)} (P_{id}^{(k)} - x_{id}^{(k)}) + c_2 r_2^{(k)} (P_{gd}^{(k)} - x_{id}^{(k)}) \quad (1)$$

$$x_{id}^{(k+1)} = x_{id}^{(k)} + V_{id}^{(k+1)} \quad (2)$$

where, i is the number of particles in the swarm; d represents the d th-dimension of particles; ω is the inertia weight coefficient; r_1, r_2 are the random values between $[0, 1]$; c_1 called cognitive factor represents belief degree on experience, which can be used to adjust the step size of particles to fly towards the direction of its local best position; c_2 known as the coefficient of social learning represents the belief degree on individuals around, which can be used to adjust the step size of particles to fly towards the direction of its global best position. The algorithm iteration termination condition is generally chosen as the maximum number of iterations or fitness

value which satisfies the predetermined threshold value of the minimum fitness after searching the optimal location.

PSO algorithm is easy to operate and simple for use with a fast convergence. However, the algorithm also has the following problems:

- Particles are "flying" toward the direction of the optimal solution. However, if the inertia factor is large, it is difficult to obtain the optimal solution and the search accuracy will reduce
- All the particles are "flying" toward the direction of the optimal solution but the closer the optimal particle comes, the less its searching speed becomes. Particle swarm deprives the diversity of solutions between the particles and thus the algorithm may converge to a local maximum without difficulty which is not always the same as the global maximum and fails to track the actual global maximum

IMPROVED PARTICLE SWARM OPTIMIZATION (IPSO) ALGORITHM

Due to later poor global search ability of PSO algorithm, the particle's flight speed is too small and thus the algorithm may converge to a local maximum. The improvement of algorithm for IPSO is as following: Under the same velocity direction, classify new particles based on the different amplitude flight. Particles with larger velocity amplitude carry out global optimization and the others carry out local optimization. Specific methods are to select individuals and global best position "extreme value" from these particles to update the particle velocity. Large velocity amplitude meets the particle global search requirements to avoid falling into local optimal and premature phenomenon; small velocity amplitude meets refine search requirements to avoid overflying the optimal solution space and it can obtain the optimum solution faster (Eberhart and Shi, 2001). Formula is shown as follows:

$$\begin{cases} V_{id}^{(k+1)} = \omega V_{id}^{(k)} + c_1 r_1^{(k)} (P_{id}^{(k)} - x_{id}^{(k)}) + c_2 r_2^{(k)} (P_{gd}^{(k)} - x_{id}^{(k)}) \\ x_{id}^{(k+1)} = x_{id}^{(k)} + V_{id}^{(k+1)} \\ V_{id}^{(1)} = b(1) V_{id}^{(0)}, \dots, V_{id}^{(m)} = b(m) V_{id}^{(0)}, \dots, V_{id}^{(j)} \\ \quad = b(j) V_{id}^{(0)}, m = 1, 2, \dots, j \\ x_{id}^{(1)} = x_{id}^{(0)} + V_{id}^{(1)}, \dots, x_{id}^{(m)} = x_{id}^{(0)} + V_{id}^{(m)}, \dots, x_{id}^{(j)} \\ \quad = x_{id}^{(0)} + V_{id}^{(j)}, m = 1, 2, \dots, j \\ |V_{id}^{(k)}| \leq V_{max} \end{cases} \quad (3)$$

where, i is the number of particles; d represents the d th-dimension of particles; ω is the inertia weight

coefficient; r_1, r_2 are the random values between $[0, 1]$; $v_{id}^{(0)}$ $n = 1, 2, \dots, J$ is called the d th-dimensional reference velocity component of particle i ; $v_{id}^{(n)}$, $n=1, 2, \dots, j$ is called the d th-dimension search velocity component of particle i , j is called speed interval number; $x_{id}^{(0)}$ is called the d th-dimension reference position component of particle i , $x_{id}^{(n)}$, $m = 1, 2, \dots, j$ is called the d th-dimension search position component of particle i ; p_{id} is the best position of particle i that has ever found; the best location found in the group is called the global best position, recorded as p_{gd} . $B(m)$, $m = 1, 2, \dots, j$, called speed coefficient, is used to determine the relationship between the search speed and the reference speed. The method to determine the relationship between the search speed and the reference speed is to set a maximum speed v_{max} and a minimum speed v_{min} . If $v_{id}^{(0)} > v_{max}$, then $b(m)$ search speed is small:

$$b_m = \frac{m}{j}$$

if $v_{id}^{(0)} < v_{min}$, then $b(m)$ search speed becomes large, $b_m = m$; if $v_{min} < v_{id}^{(0)} < v_{max}$ and when search speed is appropriate, $b(m)$ to the search speed $v_{id}^{(0)}$ of both sides becomes large or smaller:

$$b_m = 1 + \frac{m}{j}$$

Only in this way, enough solution space can be searched. The study shows that the inertia factor ω has a significant impact on optimize performance of the algorithm. Large ω will improve the convergence rate of the algorithm. While ω is small, it will improve the convergence accuracy of the algorithm. According to Eq. 4-7 the adaptive adjustment strategy for ω is proposed. That is to say, as the iteration proceeds, the value of ω should be reduced gradually.

$$\omega^n = e^{\frac{-a^n}{-a^{n-1}}} \quad (4)$$

$$a^n = \frac{1}{m} \sum_{i=1}^m |f(X_i^n) - f(X_{min}^n)|, n = 0, 1, 2, \dots \quad (5)$$

$$f(X_i^n) = f(x_{i1}^n, x_{i2}^n, \dots, x_{iD}^n) \quad (6)$$

$$f(X_{min}^n) = \min_{i=1,2,\dots,m} f(X_i^n) \quad (7)$$

where: ω^n is the inertia weight value at the n -th time iteration; D are the dimensions of the particles; $f(X_i^n)$ is the i -th particle fitness function value in the n -th generation of iteration; $f(X_{min}^n)$ is the particle fitness

function value of the location optimal particle in the n -th generation of iteration. The main characteristic of this algorithm is to linked ω with fitness function value, the fitness function value of flatness is used to dynamically adjust the ω .

The above particle swarm optimization algorithm is a particle swarm optimization with full disclosure information. Now we discuss the particle swarm optimization algorithm in the incomplete information sharing conditions. In this study, introducing the "neighborhood" concept, each particle in the particle swarm can only cognitive (or observation) information from a certain number individuals around or a certain other individuals in the area, this is the "particle swarm neighborhood". Thus the particle swarm optimization update formula is simplified:

$$\begin{cases} v_{id}^{(k+1)} = \omega v_{id}^{(k)} + c_1 \gamma_1^{(k)} (p_{id}^{(k)} - x_{id}^{(k)}) + c_2 \gamma_2^{(k)} (p_{gd}^{(k)} - x_{id}^{(k)}) \\ x_{id}^{(k+1)} = x_{id}^{(k)} + v_{id}^{(k+1)} \\ v_{id}^{(1)} = b(1)v_{id}^{(0)}, \dots, v_{id}^{(m)} = b(m)v_{id}^{(0)}, \dots, v_{id}^{(j)} \\ \quad = b(j)v_{id}^{(0)}, m = 1, 2, \dots, j \\ x_{id}^{(1)} = x_{id}^{(0)} + v_{id}^{(1)}, \dots, x_{id}^{(m)} = x_{id}^{(0)} + v_{id}^{(m)}, \dots, x_{id}^{(j)} \\ \quad = x_{id}^{(0)} + v_{id}^{(j)}, m = 1, 2, \dots, j \\ |v_{id}^{(k)}| \leq V_{max} \end{cases} \quad (8)$$

Among them: $p_{id}^{(k)}$ is the best particle position of $x_{id}^{(k)}$ in adjacent domain and it is also called neighborhood extreme; r is the random number in the interval $[0, 1]$. To new particles in Eq. 8, the following treatment is adopted: New particle preservation strategy meets the survival of the fittest, only when $x_{id}^{(k+1)}$ is better than $x_{id}^{(k)}$, it can be retained.

Consider the pros and cons of newborn particles and previous particle, the particle velocity of the particle swarm optimization algorithm can be divided into two categories: new particles generated by a velocity vector is better than the previous particle and it is called superiority particle velocity; other is called inferior particle velocity. Particle velocity update strategy is as follows: The superiority particle velocity, which can improve particle mass, is reserved and carries out global optimization search. Inferior particle velocity involves in the refinement of the search.

The analysis for basic PSO algorithm of Eq. 1 showed that the first item of the Eq. 1, i.e., $\omega v_{id}^{(k)}$ embodies the flight inertia of the particles, when a fixed value of ω is used, the balance between global search capability and the local fine search capability is poor, thus the particles can quickly fly to the current global optimum in the early running in the algorithm but when it is the most close to the global optimum, the fine search often cannot be

carried out because of the inertia of fixed flight, so that the convergence accuracy is not high. In this study, the improved particle swarm optimization algorithm uses the ω of value gradually decreases with the number of iterations, it can flexibly adjust the particle balance between global and local search capability, therefore, which both has high convergence speed in the early but also has higher convergence precision in the later.

The document Ma and Song (2009) proved that search ergodicity cannot be guaranteed in the solution space for basic PSO algorithm, thus, in theory; it cannot be guaranteed to converge to the global optimum. In this study, based on the traditional PSO algorithm, IPSO introduced speed variable coefficients and neighborhood selection algorithm to maintain the diversity of particle swarm, so ergodicity of the algorithm in the solution search space is improved, therefore, it is more likely to obtain a global optimal.

IPSO-BPNN HYBRID OPTIMIZATION ALGORITHM

Back Propagation Neural Network (BPNN) is proposed by a group of scientists headed by Rumelhart and McClelland (1986). It is a kind of multilayer feed forward network according to the error back propagation algorithm training and it is one of the most widely used neural network model at the present. Back Propagation (BP) network can learn and store a large number of mapping relationship between the input and output model. Its learning rule is to use the steepest descent method, by means of back propagation to constantly adjust the network weights and thresholds and to minimize the error sum of squares of the network.

The improved particle swarm optimization algorithm and back propagation neural network algorithm are combined together to form a hybrid algorithm. The improved particle swarm optimization algorithm is applied to calculate the back propagation neural network connection weight vector and the threshold value, namely, assume the element of the particle swarm position vector x is connection weights and threshold value between nodes of back propagation neural network, each iteration is to find weight vector and threshold value of the optimal particle. The actual output of this set of the weights vector and the threshold value are d_i ($i = 1, 2, \dots, n$; n is the number of samples).

N particles are selected to form particle swarm, where each particle is D -dimensional vector. This vector represents all the weights in the neural network model, initial value of the weight value randomly generated in the range $[0, 1]$. After the particle initial population randomly generated, subsequent the position of the each

generation particle varies according to the changes of the flight speed. The fitness value of each particle can be measured by the following formula:

$$\text{fit} = \frac{1}{n} \sum_{j=1}^n \sum_{k=1}^m (d_k - t_k)^2 \quad (9)$$

where, d_k is the actual output of the neural network, t_k is the target output, m is the output nodes, n is the number of samples for the training set.

IMPLEMENT OF IPSO-BPNN HYBRID OPTIMIZATION ALGORITHM

- Step 1:** To determine the topology of the network. According to the condition to determine the neural network input and output sample set and the topology of the network
- Step 2:** To initialization the parameters of the algorithms. The parameters of the algorithms are determined: According to Eq. 3, the position of the particle x_{id} and speed v_{id} are initialized and the number of particles N , initial inertia factor ω value, the maximum allowed number of iterations k_{max} , the acceleration coefficient c_1 and c_2 are determined, p_{id} and p_{gd} are initialized
- Step 3:** Calculating the fitness value to determine the optimal weights of neural network. Mean square error of neural network as a fitness function, each particle individual extreme value (fitness value) is calculated, the best individual extreme is selected as the global minimum and denoted p_{gd} ; the serial number of the best value particle is recorded, then the best extreme is the optimum weight of the neural network in the next iteration
- Step 4:** Calculating the fitness value and update the extreme value. The fitness value of each particle is calculated. If the value is better than the current individual extreme value of the particle, then p_{id} is set as the particle's position and update individual extreme value. If all individual particles extremum best is better than the current global extremum, then p_{gd} will be set to the particle location, the serial number of the particles is recorded and update the global minimum
- Step 5:** According to Eq. 4, the inertia weight is updated
- Step 6:** To update the position and velocity. According to the Eq. 3-7 the position and velocity are updated
- Step 7:** Test. When the iterations achieve to the maximum iteration times or meet the minimum

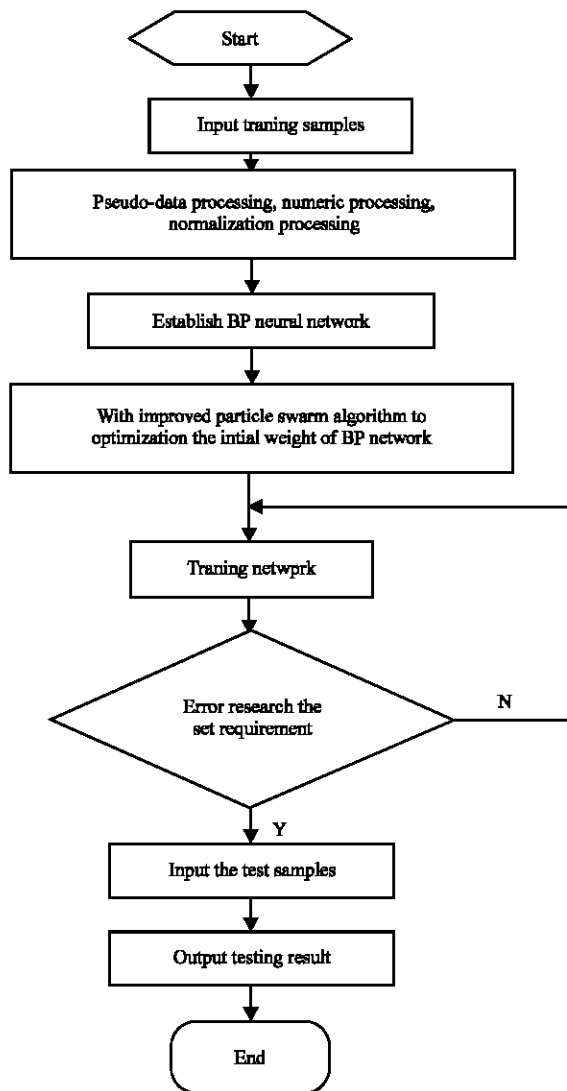


Fig. 1: Network intrusion detection process based on IPSO-BPNN

error requirement, the iteration will be stop. Connection weights and structure with the global minimum for the neural network are the optimal solution of the training problem. Otherwise, go to Step 3

Based on IPSO global optimization, a small step back propagation algorithm is used to run locally search to achieve the required precision of convergence. After the completion of the network weights training, regarding the optimal weights as the weights of BP network, network intrusion detection is conducted. In this study, the network intrusion detection process based IPSO-BPNN is shown in Fig. 1.

EXPERIMENTS AND ANALYSIS

Experimental data selection and pretreatment: In this study, we adopted the MIT KDD Cup 99 dataset to validate the effectiveness of IPSO-BPNN based network intrusion feature selection. The KDD 99 dataset contain 744 MB data with 4, 940, 000 records (University of California, 1999). Experimental data consists of four main attack types. There are:

- DoS, denial of service attacks
- U2R, unauthorized access to the local super user privileges
- Probe, scanning and probing behavior
- R2L, unauthorized access to the remote host

The data set has 41 attributes for each connection record plus one class label. If these 41 characteristics are used as the input sample feature vectors, the speed of convergence of the network is bound to be affected and the training time will inevitably increase. This article based on the 41 characteristics of their own characteristics and relevant degree connected, while taking into account the universal and the particular features, from which we select 12 features as the experimental data characteristics of the input sample. They are:

- Protocol_type, connection protocol used. Such as TCP, UDP, ICMP and other protocols, symbol type
- Service, type of service. Such as http, ftp, telnet, symbol type
- Flag, connection termination state. Flag has 11 state
- Land, the initiator address and the recipient address, symbols
- Logged_in, whether or not login is successful, Success is 1, failure to 0
- Count, the same connection record number of the target host in the past two seconds and the current. Numeric
- Srv_count, the same connection record number of the service in the past two seconds and the current. Numeric
- Dst_host_same_srv_rate, the percentage of same target host and same services. Numeric
- Dst_host_diff_srv_rate, the connection percentage of same target host and different services. Numeric
- Dst_host_same_src_port_rat, the connection percentage of same target host and same source port. Numeric
- Dst_host_srv_diff_host_rate, the connection percentage of same target host, same source port and different source host. Numeric

Table 1: Comparison of simulation results

Algo-rithm	Iteration No.	Training time (sec)	Convergence error	Detection rate (%)	False positive rate (%)
BP	164	172	0.0095417	78.65	21.35
PSO-BP	213	198	0.0009734	86.33	13.67
IPSO-BP	92	99	0.0009976	94.78	5.22

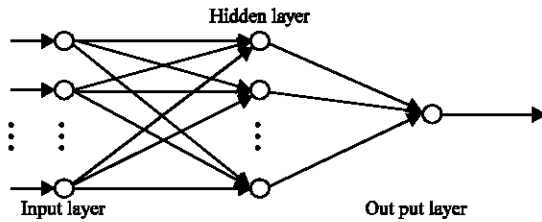


Fig. 2: Intrusion detection model of improved particle swarm optimization BP neural network

- Dst_host_error_rate, the connection percentage of same target host and SYN wrong. Numeric. Before training, the data are preprocessed to improve the quality of the input data and to help improve the convergence speed and detection accuracy
- The repeat and redundant data in the data set will be removed;
- The value of symbolic property is encoded as a numeric value;
- Due to some characteristic values in the data scope vary widely, in order to reduce the numerical difference is too large and cause problems for network training, the test set are normalized processing to the interval [0, 1]

Network structure: The network intrusion detection system based on IPSO-BPNN uses a three-tier network, as shown in Fig. 2. The nodes of the input layer depend on the feature vectors of the input sample data, so the input layer nodes is 12. The output layer is 0 indicates normal, for 1 shows abnormalities. When the number of output layer is 1, the hidden layer is a layer with 11 neurons.

Test results and analysis: The performance of intrusion detection algorithms mainly consider Detection Rate (DR) and False Positive Rate (FPR). Through these two indicators can is good enough to measure the effect of detection algorithm (Loo *et al.*, 2006). Specific expression is:

- DR** = The detected intrusion detection samples/total number of invasion samples
- FPR** = Normal samples of being false positives for the invasion/normal samples

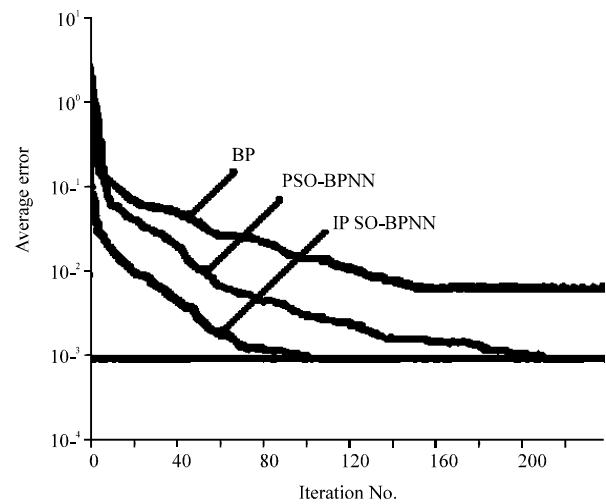


Fig. 3: Simulation results

Thousand groups' data are extracted from the data set, 400 groups of them is training data and 600 groups is test data. The number of iterations is 1600 and the target error is 0.001. Using Matlab for simulation, the simulation results are shown in Fig. 3. In order to measure the pros and cons of the proposed algorithm, this study simultaneously gives the BP intrusion detection, PSO-BPNN intrusion detection and the IPSO-BPNN intrusion detection model simulation results.

It can be seen from Fig. 3, there is a large difference between errors actually achieved and target errors using BP network intrusion detection and the error of using IPSO-BPNN algorithm intrusion detection is almost zero.

Table 1, under the same conditions, the iterations number, convergence error and training time of the IPSO-BPNN algorithm are relatively small and have a higher detection rate. In the detection of intrusion detection data, IPSO-of BPNN algorithm's performance is better than the traditional BP algorithm and PSO-BPNN algorithm.

CONCLUSION

BP neural network has the strongly nonlinear fitting ability and the learning rule is simple and easy computer, so BP learning algorithm is generally used for the training of neural network. However, the inherent defects such as easy to fall into local minimum point, slow convergence

affect the prediction accuracy and therefore subject to certain restrictions in practical applications. PSO algorithm as a swarm intelligence optimization algorithm has strong global and local search optimization capabilities. This study presents a method that combines the BP networks with the improved PSO algorithm to form hybrid IPSO-BPNN algorithm aiming at false positive rate in intrusion detection systems. And the method has applied into intrusion detection systems, which can discover the known detection exactly, forecast the new detection and reduce the invasion of omissions and the false alarm rate. Comparing the simulation results of the KDD99 CUP dataset with the intrusion detection system based on the traditional BP networks and the improved PSO-BPNN algorithm, the improved IPSO-BPNN algorithm shows less iteration times, quicker convergence rate, higher detection rate and sufficient availability.

ACKNOWLEDGMENT

This study was financially supported by the Scientific Research Fund of Hunan Provincial Science and Technology Programme, Project Number [2014FJ3157]; and Hengyang City Science and Technology Programme [2013KG63].

REFERENCES

- Clerc, M. and J. Kennedy, 2002. The particle Swarm-explosion, stability and convergence in a multidimensional complex space. *IEEE Trans. Evol. Comput.*, 6: 58-73.
- Eberhart, R.C. and Y. Shi, 2001. Particle swarm optimization: Developments, applications and resources. *Proc. Cong. Evol. Comput.*, 1: 81-86.
- Loo, E.C., M.Y. Ng, C. Leckie and M. Palaniswami, 2006. Intrusion detection for routing attacks in sensor networks. *Int. J. Distributed Sensor Networks*, 2: 313-332.
- Ma, H.F., J.F. Song and X. Yue, 2009. Hybrid neural network intrusion detection system optimized by genetic algorithm. *Commun. Technol.*, 42: 106-108.
- Mukherjee, B., L. Hebedein and K.N. Levitt, 2004. Network intrusion detection. *IEEE Network*, 8: 26-41.
- University of California 1999. KDD Cup 99 KDD dataset. <http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html>
- Van den Bergh F., 2002. An analysis of particle swarm optimizers. Department of Computer Science, University of Pretoria, South Africa.
- Wang, H., 2010. Study network intrusion detection system based on danger theory. *Computer simulation*, 27: 159-190.
- Welch, C.D.J. and M.S.D. Lathrop, 2003. A survey of wireless security threats and security mechanisms. United States Military Academy, New York, USA.
- Ye, N., S.M. Erman, Q. Chen and S. Vilbert, 2002. Multivariate statistical analysis of audit trails for host-based intrusion detection. *IEEE Transactions Computers*, 51: 810-820.
- Yi, D. and X. Ge, 2005. An improved PSO-based ANN with simulated annealing technique. *Neurocomputing*, 63: 527-533.
- Zhou, Q. and C. Wang, 2007. A number of parties method network intrusion detection based on artificial intelligence technology. *Computer Appl. Res.*, 24: 144-148.