Intrusion Feature Selection Algorithm Based on Particle Swarm Optimization

Lihong Tong, Qingtao Wu

Henan University of Science and Technology,Information Engineering College, Luoyang, 471023, China

Summary
High-dimensional intrusion detection data concentration information redundancy results in lower processing velocity of intrusion detection algorithm. Accordingly, the current study proposes an intrusion feature selection algorithm based on particle swarm optimization (PSO). Analyzing the features of the relevance between network intrusion data allows the PSO algorithm to optimally search in a featured space and autonomously select effective feature subset to reduce the data dimensionality. Experimental results show that the algorithm can effectively eliminate redundancy and reduce intrusion feature selection time to effectively increase the detection velocity of the system while ensuring detection accuracy rate.

Key words:
intrusion detection; particle swarm optimization; intrusion feature selection; optimization searching; feature relevance.

1. Introduction

Intrusion detection technology is an important research area of network security. The presence of any attack behavior against security policy in the network or system is detected by analyzing network flow or system audit record, which generates corresponding strategies, compensates system bugs, and fills system functions. However, during intrusion detection, the data size collected by the detector is large and with different features. Some of these features are not related with detection features, which reduce classification or cluster accuracy and significantly increase complicity in study, training time, and space, thereby affecting the efficiency of the algorithm operation. Therefore, methods on how to increase the detection velocity of the system without compromising the detection accuracy rate have received much attention. Previous studies reported that the intrusion feature selection could maintain the integrity of the original network data and eliminate redundancy features to increase system detection velocity. Based on current intrusion feature selection algorithms, heuristic search strategies exist, including forward search, reverse search, and sequential search, among others. Several scholars employed a method that combines expansion matrix theory and genetic algorithm, among others [1–4].

For intrusion feature selections in a large-scale datasets, these search strategies have large computing resource degradation, slow convergence rate, and high time complexity. Although the feature subsets obtained from non-search strategies in Literature [5] have relatively low time complexities, these subsets have relatively more redundancy features, which affects classification accuracy. The current study aims to improve the defects in existing algorithms by proposing an intrusion feature selection algorithm (IFSA) based on particle swarm optimization (PSO), which introduces feature relevance analysis to direct PSO with a faster convergence rate in feature space and to realize the self-adaption and self-optimization of the intrusion feature selection.

2. Feature Relevance Analysis

Relevance selection [6] is widely used in machine learning and statistics relevance analysis to evaluate relevance between features. In addition, relevance selection significantly affects the efficiency of intrusion feature selection. The relevance between two random variables is generally measured through entropy [7] defined in information theory.

Definition 1 (Entropy): For a dispersed feature X with concentrated data, the possible dereference might be $\{x_1, x_2, \cdots, x_n\}$, the corresponding probability distribution is $\{p(x_1), p(x_2), \cdots, p(x_n)\}$, and the entropy definition of X is

$$H(X) = -\sum_{i=1}^{n} p(x_i) lbp(x_i) \qquad (1)$$

Entropy expresses the information size of feature X .The smaller the entropy, the more asymmetrical the distribution of data dereferencing in X will be. The more data of some or several values X values for, the less data of other values it values. If all data of X obtain the same value, the entropy of X is 0, and the information contained in the feature set is 0; that is, no available information about such feature is present in the data set. By contrast, the bigger the entropy of X , the more data valuesdistribution of data value and the more information it contains.

Definition 2 (Joint entropy): For dispersed features X and Y in a data set, if the joint probability they value

Manuscript received December 5, 2014
Manuscript revised December 20, 2014

for x_i and y_j, respectively, is $p(x_i, y_j)$, then the Y joint entropy of X is defined as:

$$H(X,Y) = -\sum_{i=1}^{n} \sum_{j=1}^{m} p(x_i, y_j) lbp(x_i, y_j) \qquad (2)$$

The concept of entropy is derived from entropy which describes the information size shared by the two random variables. The bigger the value, the greater the relevance between the two variables will be. If joint entropy between the two variables is 0, then they are independent.

3. IFSA

IFSA employs PSO with a faster convergence rate to search in feature space. IFSA introduces relevance analysis to guide algorithm search and realizes self-adaption and self-optimization of intrusion feature selection.

3.1 Intrusion Feature Selection Definition and Principle

Definition 3 (feature subset): Feature subset is the new attribute set obtained after the irrelevant and redundant attributes entered into the attribute set are canceled.

Definition 4 (intrusion feature selection): Intrusion feature selection (also called attribute selection or feature extraction) recognizes and selects an effective attribute subset to describe an effective mode in a relatively large data set that generally contains redundant and irrelevant attributes. During feature subset selection, the algorithm generally selects the effective attribute set in the smallest scale using the following principle: (1) classification accuracy should not significantly decrease; and (2) classification distribution should maintain concord as much as possible before and after the intrusion feature selection.

3.2 PSO algorithm

Based on the research result of bird flock foraging behavior [8], Kennedy and Eberhart proposed the PSO algorithm in 1995. The PSO algorithm, with a fast execution velocity and good resistance to dimension changes, immediately attracted much attention. The algorithm is described as follows:

In a D-dimensional target search space, assuming that m particles constitute a community, in which the ith particle is expressed as a D-dimensional vector, $x_i = (x_{i1}, x_{i2}, \cdots, x_{iD})^T$ where $i = 1, 2, \cdots, m$, then the location of the ith particle in a D-dimensional search space is x_i, and the location of each particle is a potential solution. When x_i is substituted in a target function, then its adaptive value is calculated. Whether x_i is good or bad is identified and measured according to fitness. The “flight” velocity of the ith particle is also a D-dimensional vector, given by $v_i = (v_{i1}, v_{i2}, \cdots, v_{iD})^T$. To date, the optimal location of the ith particle is written as $p_g = (p_{g1}, p_{g2}, \cdots, p_{gD})^T$. PSO operates the particles demonstrated according to the following two formulas [12]:

$$v_{id}^{n+1} = \omega v_{id}^{n} + c_1 r_1 (p_{id}^{n} - x_{id}^{n}) + c_2 r_2 (p_{gd}^{n} - x_{id}^{n}) \qquad (3)$$

$$x_{id}^{n+1} = x_{id}^{n} + \eta v_{id}^{n+1} \qquad (4)$$

Where $d = 1, 2, \cdots, D; i = 1, 2, \cdots, m$; m is the colony scale; ω is the inertia weight, which is the coefficient of keeping the original velocity; c_1 is the optimal value of weight coefficient in history that particles tracking themselves, which expresses the cognition of particles to themselves. Thus, it is named “cognition” and generally set up as 2. c_2 is the weight coefficient of the optimal value that particles track the community, which expresses the particles cognition to the entire community, and is thus named “social knowledge” or usually as “society”, and is generally set up as 2. r_1, r_2 are the random numbers between [0,1]; η is the coefficient added before velocity during location update, which is named constraint factor, and is generally set up as 1; and $n = 1, 2, \cdots$ are iterations.

3.3 Binary Particle Swarm Optimization (BPSO) Algorithm

In 1997, Kennedy and Eberhart proposed the BPSO version of the PSO algorithm [10], which fueled such algorithm into a combinatorial optimization field. BPSO applies the binary coding form, and restricts each dimension x_i and p_i as 1 or 0 in BPSO model, and velocity v_i is not under such restriction. The Velocity Sigmoid function is used to express the possibility of location state change.

$$s(v) = 1/(1 + e^{-v}) \qquad (5)$$

The velocity update in Eq. (3) of BPSO is not changed. By contrast, the location Eq. (4) is changed as follows:

If ($rand() < s(v_{id}^{n})$) then $x_{id}^{n+1} = 1;$

Else $x_{id}^{n+1} = 0;$ (6)

Where $rand()$ is the random number between [0,1], and the maximum velocity $V_{\max}$ is used to restrain the possibility that x_i might be 0 or 1. The key to solve optimization problems with PSO is the fitness function selection, which embodies the relationship between practical problems and optimal algorithms.

3.4 Coding Mode

The essence of the intrusion feature selection is the selection of N features to form the subset of M features. Therefore, each feature can be defined as a one-dimensional dispersed binary variable of the particle, and the M features constitute M-dimensional binary spaces of the particles. For each particle, if the ith digit is 1, the ith feature is selected; if vice versa, then the feature is not selected. Therefore, each particle represents a different feature subset; that is, a candidate set. For example, if particle i = 100110, then features 1, 3, and 5 are selected, and the feature subset is {1,3,5}.

3.5 Fitness Function

During the intrusion feature selection process, the selection of fitness evaluation function is of prime importance. Although scholars have provided several different suggestions, such as distance evaluation and relevance evaluation, among others, a standard measurement that is generally accepted is still not available to date. The principal idea of the relevance evaluation method applied in the present study lies in selecting an attribute subset, wherein much relevance exists between each attribute and property, but no internal relevance to eliminate irrelevant and repeated attributes. The relationship between the two attributes A and B can be measured using the symmetric uncertainty, given by

$$F(A,B) = 2 * \frac{H(A) + H(B) - H(A,B)}{H(A) + H(B)} \qquad (7)$$

Relevance-based attribute selection decides goodness of an attribute set, which is measured as follows:

$$f(A,C) = \left(\sum_{j} F(A_j, C)\right) \Big/ \left(\sqrt{\sum_{i} \sum_{j} F(A_i, A_j)}\right) \qquad (8)$$

where C is the property, and i and j that include all attributes comprise the attribute set. In the fitness function of particle swarm denoted in Eq. (8), bigger values produce the higher particle fittest.

3.6 Algorithm Description

According to the above analysis, the key steps of the algorithm in the present study are as follows:

Step1: Load training data set and set initialization parameter.

Step 2: Generate original colony at random to generate random initialized velocity for each particle and set the individual extremum best of the particle and global extremum the good of the colony.

Step 3: Evaluate the fitness value of each particle based on Eq. (8).

Step 4: Compare fitness value with the best location pbest of each particle. If it is more optimal than the pbest, then it is the best current location pbest.

Step 5: Compare the fitness value of each particle with the best location gbest that the colony has ever been. If it is more optimal than the gbest, then it is the optimal location of the colony, and the reference number of the gbest is reset.

Step 6: Update particle velocity and location according to Eqs. (3), (5), and (6).

Step 7: If iterations reach the maximum value, proceed to Step 8, or repeat Step 3.

Step 8: Convert the optimal location of the colony to the corresponding feature subset and returns.

4.Experimental process and result analysis

4.1 Experimental data set

To verify the effectiveness of the algorithm, the experiment data apply KDD99 data set. The test data set collects five weeks operational data of the analog network. The data of the first and third weeks are training data with no attack cases to train the abnormal detection system. The data of the second week are test data, including 43 attack cases (64 attack methods in the entire test data) to train the misused detection system. The data on the fourth and fifth weeks are test data, including 5 million records in the data set with 41 features (attribute) in each record. The data volume in the KDD99 data set is very large; therefore, the data set needs to be sampled to decrease data volume for convenient algorithm testing. The data from the training and test data sets were randomly sampled. The randomly sampled data were combined, which formed training data subset (a total of 21836) and test data subset (a total of 36715) for experimental use. Algorithm parameters were set as follows: D = 41, m = 30, $\omega = 0.9$, $c_1 = c_2 = 2.0$, $V_{\max} = 4.0$, and 50 iterations.

4.2 Experimental programs

The following three experiments were designed to verify the performance of IFSA more clearly and correctly.

Experiment 1: The performance of the intrusion detection models based on all 41 features was compared with that of the feature subset model based on intrusion feature selection in detection time and precision. The intrusion feature selection methods proposed in the current study were first applied in the randomly sampled data set to obtain the corresponding feature subsets. Subsequently, intrusion detection models were constructed in the feature subsets based on all 41 features and after the intrusion feature selection in the training set.

Experiment 2: The performance of intrusion detection models using the proposed algorithm was compared with intrusion detection models using genetic algorithms (GA) and Relief algorithm in detection time and detection precision. The IFSA proposed in the current study was firstly applied in the data set to obtain corresponding feature subsets and construct intrusion detection models based on the feature subset. The GA and Relief algorithms were then used for the intrusion feature selection to the same training data set.

Experiment 3: Support vector mechanism (SVM) was used as the classifier. The classification error score generated by sample classification of the three feature subsets were compared after the intrusion feature selection in Program 2.

4.3 Experiment Result Analysis

The IFSA algorithm in the current study was applied in the experimental training data set after random sampling. Form 1 demonstrates the generated feature subset.

Form 1 Feature Subset Obtained via the IFSA Algorithm

Attack type	***Feature subset***
Dos	Dos, protocol_type, src_bytes, count, dst_host_same_src_rate
PROBE	Probe, duration, service, src_bytes, dst_bytes, count, dst_host_diff_src_rate
R2L	R2L, duration, service, src_bytes
U2R	U2R, duration, service, src_bytes, root_shell, dst_host_count
NORMAL	protocol_type, service, src_bytes, count, dst_host_count

Form l shows that different feature subsets for different attacks were obtained after the intrusion feature selection. For each attack type, data sets for intrusion detection model were constructed. These data sets have not gone through the intrusion feature selection and feature subsets after intrusion feature selection demonstrated in Form 1. The performance of each intrusion detection model in detection time and detection precision was compared. The results from the comparison are shown in Form 2.

Form 2 Comparison in Detection Rate and Detection Time Before and After the Intrusion Feature Selection

Attack type	***Classification precision***		***Detection Time (Sec)***	
	All features	***Feature subset***	***All feature***	***Feature subset***
Dos	83.5%	98.4%	1.09	0.25
PROBE	87.4%	97.5%	1.23	0.36
R2L	85.7%	98.7%	1.16	0.28
U2R	84.2%	97.1%	0.94	0.17
NORMAL	86.3%	98.8%	1.13	0.24

Form 2 shows that the performance of intrusion detection models using the algorithm proposed in the current study is obviously better than the intrusion detection models that have not gone through the intrusion feature selection in detection precision and detection time. IFSA, GA, and Relief algorithms were used in the training data subset for experimental use after random sampling. The experimental results are presented in Forms 3 and 4.

Form 3 Comparison of Classification Precision Among IFSA, GA, and Relief algorithms

Attack type	***Classification precision***		
	IFSA	***GA***	***Relief***
Dos	98.4%	97.3%	95.6%
PROBE	97.5%	96.7%	96.4%
R2L	98.7%	98.9%	94.7%
U2R	97.1%	95.3%	95.3%
NORMAL	98.8%	97.2%	95.8%

Form 4 Comparison among IFSA, GA, and Relief Algorithms in Classification Detection Time

Attack type	***Detection Time (Sec)***		
	IFSA	***GA***	***Relief***
Dos	0.25	0.35	0.41
PROBE	0.36	0.43	0.48
R2L	0.28	0.38	0.47
U2R	0.17	0.21	0.33
NORMAL	0.24	0.36	0.42

Finally, SVM was used as the classifier. Feature subsets obtained after the IFSA, GA, and Relief algorithms performed an intrusion feature selection of the experimental training data set as the classification sample. The experimental results are demonstrated in Forms 5 and 6.

Form 5 IFSA, GA, and Relief Classification Precision Comparison using the SVM Classifier

Data Set	***Classification Precision***		
	IFSA	***GA***	***Relief***
Training subset	98.4%	97.3%	95.6%
Experimental subset	97.5%	96.7%	96.4%

Form 6 IFSA, GA, Relief detection time Comparison using the SVM Classifier

Data set	***Detection Time (Sec)***		
	IFSA	***GA***	***Relief***
Training subset	0.25	0.35	0.41
Test subset	0.46	0.59	0.53

Form 4 shows that no significant difference exists among the feature subsets as classification sample generated through IFSA algorithm and the basic GA algorithm proposed in the current study in terms of classification precision, which are higher than the Relief algorithm in precision rate. In terms of intrusion feature selection time, the method proposed in the present study exhibited the best performance.

Combining the results of Experiments 1, 2, and 3 reveals that the IFSA algorithm effectively reduces the feature dimension of data information. In addition, the performance of intrusion detection models based on IFSA algorithm is better than intrusion detection models that have not gone through the intrusion feature selection in detection time and detection precision. Compared with GA and Relief algorithms, IFSA algorithm can significantly reduce the time complexity of intrusion feature selection under high classification precision and effectively shorten intrusion feature selection time.

5. Conclusions

This paper proposes the algorithm IFSA based on PSO. By analyzing the relevance among all the features in network intrusion data, IFSA uses the optimization searching of PSO algorithm in all feature spaces, conducts guided search according to relevance, and select effective feature subsets through self-adaptation and self-optimization to reduce the data dimensionality. The result of intrusion detection models for the experimental verification to KDD99 data set shows that the intrusion feature selection method applied in the current study ensure the detection accuracy of the premise to effectively improve the system detection performance. Therefore, IFSA is better than the performance of existing intrusion feature selection methods in detection time and classification accuracy.

Acknowledgments

The authors would like to thank the reviewers for their detailed reviews and constructive comments, which have helped improve the quality of this paper. This work was supported in part by the Plan for Scientific Innovation Talent of Henan Province under Grant No. 124100510006.

References

[1] Sivatha Sindhu S S, Geetha S, Kannan A. Evolving optimised decision rules for intrusion detection using particle swarm paradigm[J]. International Journal of Systems Science, 2012, 43(12): 2334-2350.

[2] Boughaci D, Kadi M D E, Kada M. Fuzzy particle swarm optimization for intrusion detection[C]//Neural Information Processing. Springer Berlin Heidelberg, 2012: 541-548.

[3] Chung Y Y, Wahid N. A hybrid network intrusion detection system using simplified swarm optimization (SSO)[J]. Applied Soft Computing, 2012, 12(9): 3014-3022.

[4] Jiang F, Ling S H, Chan K Y, et al. An immunology-inspired multi-engine anomaly detection system with hybrid particle swarm optimisations[C]//Fuzzy Systems (FUZZ-IEEE), 2012 IEEE International Conference on. IEEE, 2012: 1-8.

[5] Aljarah I, Ludwig S A. MapReduce Intrusion Detection System based on a Particle Swarm Optimization Clustering Algorithm[C]. 2013 IEEE Congress on Evolutionary Computation, CEC 2013 ,2013,955-962.

[6] Liu Y, Ma R. Network anomaly detection based on BQPSO-BN algorithm[J]. IETE Journal of Research, 2013, 59(4): 334-342.

[7] Sailaja M, Kumar R K, Murty P S R, et al. A Novel Approach for Intrusion Detection Using Swarm Intelligence[C]//Proceedings of the International Conference on Information Systems Design and Intelligent Applications 2012 (INDIA 2012) held in Visakhapatnam, India, January 2012. Springer Berlin Heidelberg, 2012: 469-479.

[8] Satpute K, Agrawal S, Agrawal J, et al. A Survey on Anomaly Detection in Network Intrusion Detection System Using Particle Swarm Optimization Based Machine Learning Techniques[C]//Proceedings of the International Conference on Frontiers of Intelligent Computing: Theory and Applications (FICTA). Springer Berlin Heidelberg, 2013: 441-452.

[9] Sheikhan M, Sharifi Rad M. Using particle swarm optimization in fuzzy association rules ‐ based feature selection and fuzzy ARTMAP ‐ based attack recognition[J]. Security and Communication Networks, 2013, 6(7)797-811.

[10] Wang L, Zhang S, Li Y, et al. An Attribute-weighted Clustering Intrusion Detection Method[J]. Journal of Networks, 2013, 8(6): 1278-1284.