Journal of Harbin Engineering University Vol 44 No. 7
ISSN: 1006-7043 July 2023

Intrusion Detection Using Improved Drosophila Optimization Based Weighted Extreme Learning Machine

P. Kaliraj[1]
pkalirajmsc@gmail.com
Assistant Professor and Research Scholar, Kongunadu Arts and Science College,
Coimbatore, Tamil Nadu, India,

Subramani. B[2]
drbsubramani@gmail.com
Principal and Research Supervisor, SNMV College of Arts and Science,
Coimbatore, Tamil Nadu, India,

Abstract: With the rapid development of computer networks, network attacks and damages are becoming more and more frequent, and network intrusion detection has become a hot research topic. With the rapid development of artificial intelligence technology in recent years, more and more researchers apply artificial intelligence technology to network intrusion detection. Extreme Learning Machine (ELM) is a feedforward neural network training method. It randomly sets the initial weights of the feedforward neural network, and completes the network training by solving the least squares solution of the output weights. In this research, the weights evaluated during the machine learning process is optimized using Improved Fruit Fly algorithm. This improves the learning speed and generalization performance of Extreme Learning Machine. To evaluate the performance of the proposed algorithm NSL-KDD dataset is used. Experimental results show that the algorithm proposed can effectively improve the recognition rate of intrusion detection and reduce the rates of false positives and false negatives.

Keywords: Intrusion Detection, Single Hidden Layer Feedforward Neural Network Model, Extreme Learning Machine, Fruit Fly Optimization.

1. INTRODUCTION

The research of machine learning in intrusion detection has been highly concerned by researchers in the field of network security, and has achieved certain results in its theoretical development, key technology research and application [1-2]. The main machine learning algorithms used in intrusion detection are: Bayeux the classification algorithm [3], the support vector machine [4-5] and BP Neural Networks Algorithms [6-8], etc. Reference [9] proposes an effective cooperative intrusion detection. In this network, each intrusion detection system uses Bayesian learning to evaluate the intrusion detection rate and false positive rate of neighbouring systems, and uses a Bayesian decision model to cluster the results to minimize errors. The cost of intrusion decision and maintenance. Reference [10] proposed a parallel intrusion detection model that first uses 'Relief'the algorithm reduces the data, and then uses the improved crow search algorithm to perform the dimensionality reduction operation of the data and the kernel extreme learning machine in parallel. The parameters are optimized, and finally the trained kernel extreme learning machine is used to Classification of the test set. The results show that the model can greatly improve the classification accuracy and speed of intrusion detection. In the literature [11], a novel Single Hidden Layer Feedforward Neural Network Model Extreme Learning Machine (ELM) was mentioned by Huang Guangbin et al. out, this model is compared to the traditional BP Neural network does not need reverse iteration It has the advantages of relatively short learning time and good generalization performance. The literature [12] is about BP Neural network algorithms are hacking the problems of slow convergence and low detection rate in detection problem, proposed a method based on Principal Component Analysis (PCA) and Extreme Learning Machine (ELM) intrusion detection algorithm. Compared with the traditional machine learning algorithm, the algorithm has

the advantages of intrusion detection rate, correct rate, false negative rate and False alarm rate and other evaluation indicators have improved.

Since ELM was proposed, it has received more attention, and it has also shown good performance in intrusion detection [13-14]. The problem is that when the distribution of data types is not balanced, the effect of this method on classification will be greatly reduced. decline. In the existing machine learning algorithms for intrusion detection, the problem of data imbalance in network traffic is less considered.In this paper, an improved fruit fly algorithm is proposed to optimize the weighted extreme learning machine intrusion detection algorithm. The algorithm uses the weighted extreme learning machine [15] as a feedforward neural network, which has the advantages of short training time and good generalization performance. The imbalance in the NSL-KDD intrusion detection data set increases the weight of minority attacks, so that the detection rate of minority attacks in network attacks is greatly improved compared to traditional machine learning. The optimization ability is used to globally optimize the input weights and biases of the hidden layer in the weighted extreme learning machine, so as to avoid the algorithm from falling into the local optimal solution, and realize the classification of the NSL-KDD intrusion detection data set.

2. BACKGROUND OF WEIGHTED EXTREME LEARNING MACHINE

Based on ELM, this paper establishes a weighted extreme learning machine classification model. The network structure of the ELM model is shown in Figure 1.In Figure 1, suppose given N training samples$\{x_i, t_i\}_{i=1}^{N}$, $x_{i=[x_{i1},x_{i2},\ldots,x_{in}]^T} \epsilon R^n$, $t_{i=[t_{i1},t_{i2},\ldots,t_{in}]^T} \epsilon R^m$,where n is the number of features of the sample, m is the number of categories of the sample. A feedforward neural network output model with L hidden layer nodes can be expressed as follows:

$$\sum_{h=1}^{L} \beta_h G(a_h, b_h, x_i) = t_{i,i=1,2,\ldots,N}$$

where: $\beta_h$is the output weight of the h^{th} hidden layer neuron; G is the activation function of the hidden layer neurons; a_h , b_h respectively h individual Input weights and biases of hidden layer neurons; x is the input sample, o_i for the first The actual output value of i training samples ; t_i for the first Expected output for i training samples.

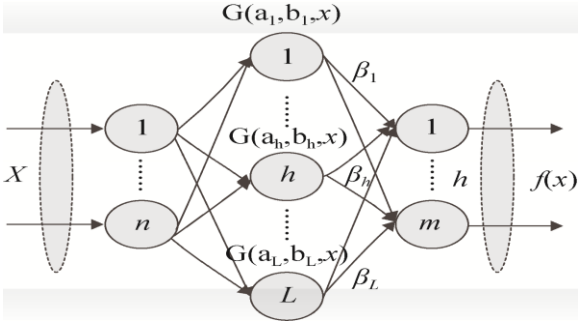

Figure1.BasicStructureDiagramofSingleHiddenLayerFeedforwardNeuralNetwork

According to [15], for a quantity of N training samples, $\{x_i, t_i\}_{i=1}^{N}, x_i \epsilon R^n$ There exists a (a_b,b_h) and β_h with $\sum_{i=1}^{L} \|o_i - t_i\| = 0$, so that the single-hidden layer feedforward neural network model can approximate the training set with zero error $\{x_i, t_i\}_{i=1}^{N}, x_i \epsilon R^n$ which is

$$\sum_{h=1}^{L} \beta_h G(a_h, b_h, x_i) = t_{i,i=1,2,\ldots,N} \ldots\ldots (1)$$

The equation can be further simplified as: Hβ=T

where H is the output matrix of the hidden layer; β is the output weight of the hidden layermatrix; T is the expected output matrix corresponding to the training samples.

During the training process of ELM, when initializing the network parameters, the input weights ah of the hidden layer and the bias bh of the hidden layer are randomly generated, and remain unchanged during the whole training and testing process. Since the input training samples, the input weights and biases of the hidden layer, and the expected output are all known, the entire training process is to obtain the hidden layer output weight matrix β in the ELM model, so as to obtain a complete classification model.

From the Moore-Penrose generalized inverse matrix H+ of the hidden layer output matrix H, we can get

$$\hat{\beta} = H^{+}T \ldots\ldots (2)$$

In the formula: There are many ways to calculate H+. In ELM, the orthogonal projection method (KKT) is usually used to solve H+. When

H^T H is a non-singular matrix, $H^+=(H^TH)^{-1}H^T$; when HH^T is a non-singular matrix, $H^+=H^T(HH^T)H^{-1}$.

In order to solve Equation (2), a sufficiently small regular term 1/C is added to the diagonal of HH or HH^T, so that the classification model has better stability and generalization performance. The output weight of the hidden layer can be expressed as,

$$\hat{\beta} = \begin{cases} H^T(I/C + HH^T)^{-1}T, & N < 1 \\ (I/C + H^TH)^{-1}H^TT, & N \geq 1 \end{cases}$$

The output function of ELM can be expressed as,

$$f(x) = h(x)\hat{\beta} = \begin{cases} h(x)H^T(I/C + HH^T)^{-1}T, & N < 1 \\ h(x)(I/C + H^TH)^{-1}H^TT, & N \geq 1 \end{cases} \dots\dots(3)$$

In the classification problem, not all classified sample data are evenly distributed. In order to solve the classification problem of unbalanced samples, Zong et al. [15] proposed a Weighted Extreme Learning Machine (WELM) algorithm based on ELM. Reference [16] proposes to assign weights to each sample according to the weighting scheme:

Weighting scheme one W1: Automatic weighting scheme:

$$W1 = \frac{1}{Count\ (t_i)}$$

where Count(t_i)is the number of samples of class t_i in the training sample.

Weighting scheme two W2: push the ratio of the minority class to the majority class to the direction of 0.618:1 (the golden ratio). This plan actually sacrifices the classification accuracy of the majority class in exchange for the classification accuracy of the minority class.

$$W2 = \begin{cases} \frac{0.618}{Count(t_i)}, t_i\ belong\ to\ the\ majority\ class \\ \frac{1}{Count\ (t_i)}, t_i\ belong\ to\ the\ minority\ class \end{cases}$$

The output weights of the WELM hidden layer can be expressed as,

$$\hat{\beta} = H^+T = \begin{cases} H^T(I/C + WHH^T)^{-1}WT, & N < 1 \\ (I/C + H^TWH)^{-1}H^TWT, & N \geq 1 \end{cases}$$

In the formula: the weighting matrix is a diagonal matrix of NXN; the N main diagonal elements correspond to N samples, and different weights are assigned to different sample categories, and the weighted weights of the same category are the same.

When the hidden layer feature map h(x) is unknown, the kernel matrix is defined as,

$$\Omega_{ELM} = HH^T: \Omega_{ELMi,j} = h(x_i)h(x_j) = K(x_i, x_j) \dots\dots(4)$$

In the formula: Ω_{ELM} is the kernel matrix, and the kernel function K needs to satisfy the Mercer condition. Common kernel functions include Gaussian kernel function, radial basis kernel function, polynomial kernel function and linear kernel function. By formula (4), the output function expression (3) can be expressed as,

$$f(x) = h(x)\hat{\beta} = h(x)H^T\left(\frac{I}{C} + WHH^T\right)^{-1}WT = \begin{bmatrix} K(x, x_1) \\ \cdot \\ \cdot \\ \cdot \\ K(x, x_n) \end{bmatrix}(I/C + W\Omega_{ELM})^{-1}WT \dots\dots(5)$$

Therefore, the training process of the classification model based on the weighted extreme learning machine is:

(1) Randomly set the input weights ah and bias bh of the hidden layer, where h=1,2, ...,L;

(2) According to the weighting scheme, assign weights to each sample, and calculate the weighting matrix W;

(3) Calculate the kernel matrix Ω_{ELM} according to the selected kernel function;

(4) Calculate the output using equation (5).

2. A WEIGHTED EXTREME LEARNING MACHINE ALGORITHM BASED ON IMPROVED FRUIT FLY OPTIMIZATION

2.1 Improved Drosophila Optimization Algorithm

Inspired by the foraging behavior of Drosophila, Pan Wenchao proposed the Fruit Fly Optimization Algorithm (FOA) [16]. The basic optimization process can be divided into the following steps [17-18]:

Step 1: Parameter initialization, set the population size N, the maximum number of iterations max and the fruit fly group position Xaxis, Yaxis give the random direction and distance of each fruit fly individual, and then the fruit fly individual starts to use smell to search for food :

X_i= Xaxis + Rand()

Y_i= Yaxis + Rand()... ... (6)

In the formula: Rand() is the flight range of the fruit fly, that is, the iterative step size.

Step 2: Calculate the distance $Dist_i$ between each fruit fly individual and the coordinate origin, and then calculate the taste concentration judgment value Si of each fruit fly individual:

$$Dist_i = \sqrt{X_i^2 + Y_i^2}$$

$$S_i = 1/Dist_i$$

Step 3: Substitute the taste concentration judgment value Si in step 2 into the fitness function (Fitness Function) to find the taste concentration Smelli of each individual location of the fruit fly, and find the fruit with the best taste concentration in the fruit fly population Fly (for maximum value):

$Smell_i$= Function(S) … … (7)

[bestSmell bestIndex] = max($Smell_i$)

Step 4:Record the taste concentration value of the fruit fly with the best taste concentration and its position coordinates. At this time, the fruit fly group exerts its visual advantage.

Gradually fly towards this position:

Smellbest = bestSmell

Xaxis = X(bestIndex)… … (8)

Yaxis = Y(bestIndex)

Step 5: Enter the iterative optimization stage, repeat the above steps 2~3, and determine whether the taste concentration value is greater than the taste concentration of the previous iteration. If not, continue to repeat the above steps 2~3 within the maximum number of iterations; if so, go to step 4.

In the actual application process, the search distance of the fruit fly optimization algorithm is usually a fixed value, which lacks adaptability, which leads to the decline of the global search ability of FOA, and it is easy to fall into local search [19]. In order to solve the shortcomings of the fruit fly optimization algorithm, this paper makes corresponding improvements to the search distance of the fruit fly optimization algorithm. In the early search process of Drosophila, a maximum range of search radius is first given, which is convenient to quickly find a better solution in the global range. A more precise search will finally determine the optimal solution. Therefore, this paper improves formula (6), as shown in formula (9):

X_i= Xaxis + a^kx Rand()

Y_i= Yaxis + a^kx Rand()… … (9)

where α is the step size control factor [20]; k is the number of iterations. As the number of iterations increases, the search range of the fruit fly will decrease exponentially.

2.2 Intrusion Detection and Classification Algorithm Based on IDOA-WELM

In this paper, when WELM is used as a feedforward neural network, it has the advantages of short training time, good generalization performance, and the global optimization ability of IDOA. The IDOA algorithm is used to optimize the randomly determined hidden layer input weights and biases in the WELM network. Processing, not only solves the problem of data imbalance in network intrusion detection, greatly improves the recall rate of minority attacks in network attacks, but also avoids WELM from falling into local optimal solutions. The algorithm flow of this paper is shown in Figure 2.

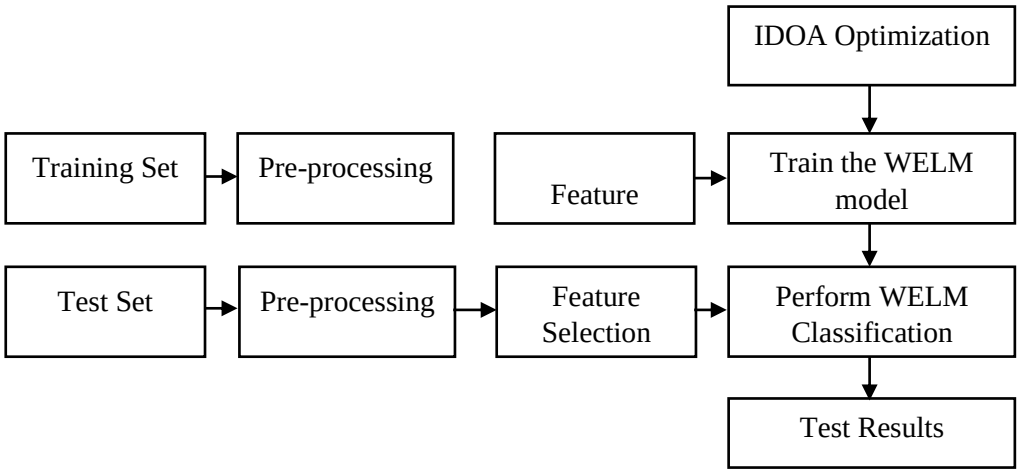

Figure 2. Intrusion Detection Model using Weighted ELM Classification

The specific algorithm flow is as follows:
(1) Select the training set and test set from the NSL-KDD dataset, and preprocess the data in the dataset;
(2) Analyze the correlation between each feature in the data set, and select some features with greater correlation to reduce the dimension of the data set;
(3) Initialize the network parameters of IDOA and WELM: randomly determine the initial position [Xaxis, Yaxis] of the fruit fly population, set the population size N, the step size control factor α and the maximum number of iterations max, the input neurons and hidden layers of WELM The number of neurons and output neurons, determine the weighting scheme, and initialize the input weights and biases of the WELM hidden layer;
(4) Input the training set into WELM, and calculate the taste concentration of individual fruit flies according to formula (7);
(5) Find the fruit fly individual with the best taste concentration in the fruit fly population, and record its position according to formula (8);
(6) Update the position and flight direction of the fruit fly according to formula (9), and enter the iterative optimization stage;
(7) If the number of iterations is greater than max, save the position of the fruit fly individual with the optimal taste concentration, that is, the input weight and bias of the globally optimal hidden layer; otherwise, increase the number of iterations by one and return to step (2) ;
(8) Substitute the optimal hidden layer input weights and biases into WELM, and conduct experiments on the test set.

3 ALGORITHM SIMULATION AND RESULT ANALYSIS

3.1 Data set selection and preprocessing

In this paper, the NSL-KDD data set [21] is selected as the experimental data set, and KDDTrain+ and KDDTest+ in the NSL-KDD data package are selected as the training set and test set of the experiment respectively. Each dataset has 42-dimensional data, of which the first 41 dimensions are dataset features, and the 42nd dimension is dataset label bits. The tag bits include normal data Normal and 39 types of attacks, of which the 39 types of attacks belong to four types of attacks: DOS, U2R, R2L and Probe. Among them, the training set includes 21 types of intrusion attacks, and there are 18 types of intrusion attacks in the test set. These intrusion attacks only appear in the test set, which can be used to evaluate the detection ability of the intrusion detection algorithm in this paper against unknown attacks. Table 1 shows the distribution of each label class in the training set and test set.

Table 1. Number of Features in Each subclass in Training and Test Set

Class	Normal	DOS	U2R	R2L	Probe	Total
Training Set	67343	45927	52	995	11656	125973
Test Set	9711	7458	200	2754	2421	22544

Before training the model, the data in the dataset needs to be preprocessed:
(1) Convert the character type in the feature into the 41-dimensional feature of the numerical NSL-KDD dataset, the second-dimensional feature protocol type (Protocol Type), the third-dimensional feature network service (Service) and the fourth-dimensional feature connection The state (Flag) is a character type feature and needs to be converted into a numerical type. Denote TCP as 1, UDP as 2, and TCMP as 3 in the second dimension feature. The 67 service types in the third dimension feature are respectively recorded as 1~67 in the alphabetical order of their names. In the fourth dimension feature,the 11 kinds of Flag states are recorded as 1~11 respectively. In the 42nd dimension label bits, there are 5 types of labels: Normal, DOS, Probe, U2R and R2L, which are recorded as 0~4 respectively.
(2) Data standardization and normalization
The numerical data set in the previous step is processed according to formula (10) and formula (11), and the measurement units between different eigenvalues are unified to

reduce the detection results caused by the difference of measurement units. influence.

$$\text{Normalized formula: } x_1 = \frac{x - \bar{x}}{\sigma} \ldots\ldots (10)$$

In the formula: x is the eigenvalue; $\bar{x}$ is the mean value of the eigenvalue; σ is the standard deviation of the eigenvalue; x1 is the standardization result of the dimension of each data sample.

The normalization formula is:

$$\text{Normalized formula: } x_2 = (x_1 - x_{1min})/(x_{1max} - x_{1min}) \ldots\ldots (11)$$

In the formula: x_{1min} is the minimum value of all samples of this dimension feature processed by formula (10); x_{1max} is the maximum value of all samples of this dimension feature processed by formula (10); x_2 is the value of each data sample. The result after dimension feature normalization.

3.2 Performance Measures

In this paper, the three performance evaluation indicators of precision rate, false alarm rate and recall rate are used to evaluate the advantages and disadvantages of the algorithm in this paper.

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN}$$

$$False\ Positive\ Rate(FPR) = \frac{FP}{FP + TN}$$

$$Recall = \frac{TP}{TP + FN}$$

where TP (True Positive) example: the actual intrusion attack, the predicted result is also the data volume of the intrusion attack (the number of samples correctly identified as normal data); TN (Ture Negative) example: the actual normal data , the prediction result is also the data volume of normal data (the number of samples correctly identified as attack data); FP (False Positive) example: the actual data is normal data, and the predicted result is the data volume of intrusion attacks (wrongly identified as normal data) The number of data samples); FN (False Negative) example: the actual intrusion attack, the predicted result is the data volume of normal data (the number of samples that are wrongly identified as attack data).

3.3 Data Dimensionality Reduction Processing

The pre-processed 41-dimensional data features are analysed for correlation by the method of Pearson correlation coefficient, and the weight ranking diagram of each feature is obtained, in which the ranking order is arranged according to the order of the features in the NSL-KDD data set, as shown in Figure 3.

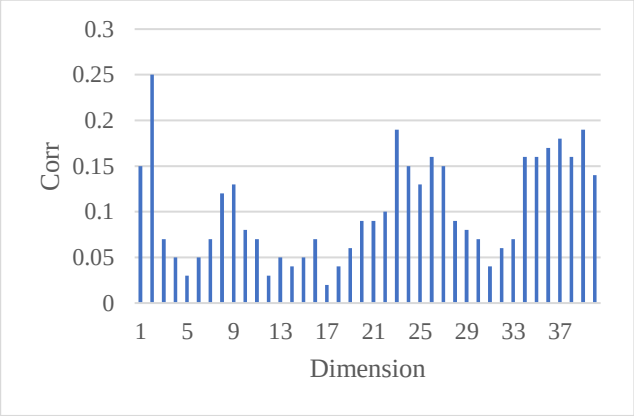

Figure 3. Sorting Graph of Feature Weights

In the experiment, the features with correlation coefficient (corr)>0.05 were taken as the training set and test set after dimension reduction. Therefore, the dimensionality-reduced training set and test set contain 18-dimensional features.

3.4 Simulation and Result Analysis

The simulation environment of this paper is the inter i7 processor 3.2GHz, 8GB memory, which is realized by compiling a simulation program using MATLAB.

(1) Comparison experiment between WELM and other machine learning algorithms

The number of WELM input layer nodes is set to 18, the number of hidden layer nodes is 250, the number of output layer nodes is 5, and the weighting scheme selection method is

Case 2, the kernel function selects the Sigmoid kernel function. The detection results of the algorithm in this paper are compared with the detection results of K-nearest neighbor algorithm (KNN), random forest (Random Forest, RF), BP neural network and extreme learning machine (ELM) algorithm. The comparison test results are shown in Table 2.

It can be seen from Table 2 that the weighted extreme learning machine algorithm is almost the same as other machine learning algorithms

in the detection effect of Normal, Dos and Probe, and in the detection effect of U2R and R2L, the recall rate of WELM is Much higher than other machine learning algorithms. But in contrast, the recall rates of these two types are still relatively low, because the two major attacks, U2R and R2L, appear less in real life, so the sample size in the training set is too small, and the data set is unbalanced. Phenomenon.

(2) Comparative experiment of WELM and IDOA-WELM algorithm

The WELM algorithm optimized by IDOA and FOA respectively is compared with the WELM algorithm without optimization. The population size of the fruit fly is set to 50, the number of iterations is 300, and the direction and distance of the fruit fly are set to [–20, 20], the step size control factor α is 0.95. The number of nodes in the input layer of WELM is 18, the number of nodes in the hidden layer is 250, and the number of nodes in the output layer is 5. The weighting scheme selects scheme 2, and the kernel function selects the Sigmoid kernel function. The experimental results are shown in Table 2.

Table 2. Comparison table of Detection Results of Different Algorithms

Detection Model	Recall (%)					Accuracy (%)	False Alarm Rate (%)	Detection Times(s)
	Normal	DOS	U2R	R2L	Probe			
KNN	98	77	35	12	64	82	35	3.2
BPN	96	75	37	25	63	82	33	3.2
SVM	98	74	34	46	67	83	33	3.1
ELM	97	78	32	47	62	84	28	2.5
WELM	98	79	44	45	68	86	13	2.3
FOAELM	98	78	45	47	68	88	8	1.8
IDOAELM	98	79	45	48	67	88	3	1.6

It can be seen from Table 2 that the WELM algorithm optimized by FOA or IDOA has undergone global optimization, which improves the recall rate of the four types of attacks. Compared with FOA-WELM, IDOA-WELM has better classification results, especially the recall rate of U2R attack is increased by 6% compared with WELM, the test set classification accuracy rate is increased by 3%, and the false alarm rate is reduced by 4.5%. Under the same experimental environment, the population size of the fruit fly is set to 50, the number of iterations is 300, the direction and distance of the fruit fly are set to [–20, 20], and the step size control factor α is set to 0.95. The training time of FOA-WELM is 1.8 s, and the training time of IDOA-WELM algorithm is 1.6 s. This is because the adaptiveness of the newly added optimization algorithm increases the time complexity, so the training time increases.

4. CONCLUSION

Aiming at the problem of data imbalance in intrusion detection, this paper uses the WELM algorithm to increase the detection weight of the minority class, and at the same time, uses the FOA algorithm to optimize it, which improves the recall rate of the minority class in the intrusion detection to a certain extent. The simulation results It is shown that compared with the traditional machine learning algorithm, the recall rate of the WELM algorithm for the two major minority attacks is improved to a certain extent, and the false alarm rate is reduced accordingly. Therefore, WELM is more suitable for the research of intrusion detection; using the IDOA algorithm The optimized WELM algorithm further improves the recall rate and false alarm rate, reduces the training time, and improves the real-time performance of intrusion detection.

REFERENCES

[1] Chakravarthy, S. S., & Rajaguru, H. (2022). Automatic detection and classification of mammograms using improved extreme learning machine with deep learning. https://doi.org/10.1016/j.irbm.2020.12.004

[2] Ahmad, Z., Shahid Khan, A., Wai Shiang, C., Abdullah, J., & Ahmad, F. (2021). Network intrusion detection system: A systematic study of machine learning and deep learning approaches. *Transactions on Emerging Telecommunications Technologies*, *32*(1), e4150. https://doi.org/10.1002/ett.4150

[3] Ahmim, A., Maglaras, L., Ferrag, M. A., Derdour, M., & Janicke, H. (2019, May). A novel hierarchical intrusion detection system based on decision tree and rules-based models. In *2019 15th International Conference on Distributed Computing in Sensor Systems (DCOSS)* (pp. 228-233). IEEE. https://doi.org/10.1109/DCOSS.2019.00059

[4] Zhang, H., Li, Y., Lv, Z., Sangaiah, A. K., & Huang, T. (2020). A real-time and ubiquitous network attack detection based on deep belief network and support vector machine. *IEEE/CAA Journal of Automatica Sinica*, *7*(3), 790-799. https://doi.org/10.1109/JAS.2020.1003099

[5] Gu J, Wa ng L H, Wa ng H W , et al. A Novel Approach to Intrusion Detection Using SVM Ensemble with Feature Augmentation[J]. Computers and Security (S0167-4048), 2019, 86(9): 53-62.

[6] Hajimirzaei, B., & Navimipour, N. J. (2019). Intrusion detection for cloud computing using neural networks and artificial bee colony optimization algorithm. *Ict Express*, *5*(1), 56-59. https://doi.org/10.1016/j.icte.2018.01.014

[7] Ding Hongwei, Wan Liang, Deng Xuankun . Improved HS Algorithm optimization BP nerveResearch on network intrusion detection [J]. Computer Engineering and Science, 2019, 41(1): 65-72.

[8] Yang, A., Zhuansun, Y., Liu, C., Li, J., & Zhang, C. (2019). Design of intrusion detection system for internet of things based on improved BP neural network. *Ieee Access*, *7*, 106043-106052. https://doi.org/10.1109/ACCESS.2019.2929919

[9] Cao, X., Fu, Y., & Chen, B. (2020). Packet-based intrusion detection using Bayesian topic models in mobile edge computing. *Security and Communication Networks*, *2020*, 1-12. https://doi.org/10.1155/2020/8860418

[10] Ma Chao. Network Intrusion Based on ReliefF and Improved Crow Search Optimization Parallel Method[J]. Journal of Application Research of Computers, 2019, 36(10): 3063-3068.

[11] Dwivedi, V., & Srinivasan, B. (2020). Physics informed extreme learning machine (pielm)–a rapid method for the numerical solution of partial differential equations. *Neurocomputing*, *391*, 96-118. https://doi.org/10.1016/j.neucom.2019.12.099

[12] Gao, J., Chai, S., Zhang, B., & Xia, Y. (2019). Research on network intrusion detection based on incremental extreme learning machine and adaptive principal component analysis. *Energies*, *12*(7), 1223. https://doi.org/10.3390/en12071223

[13] Chen, J., Qi, X., Chen, L., Chen, F., & Cheng, G. (2020). Quantum-inspired ant lion optimized hybrid k-means for cluster analysis and intrusion detection. *Knowledge-Based Systems*, *203*, 106167. https://doi.org/10.1016/j.knosys.2020.106167

[14] Logeswari, G., Bose, S., & Anitha, T. (2023). An intrusion detection system for sdn using machine learning. *Intelligent Automation & Soft Computing*, *35*(1), 867-880. doi: 10.32604/iasc.2023.026769

[15] Xu, Z., Liu, J., Luo, X., Yang, Z., Zhang, Y., Yuan, P& Zhang, T. (2019). Software defect prediction based on kernel PCA and weighted extreme learning machine. *Information and Software Technology*, *106*, 182-200.https://doi.org/10.1016/j.infsof.2018.10.004

[16] Zainel, Q. M., Darwish, S. M., & Khorsheed, M. B. (2022). Employing Quantum Fruit Fly Optimization Algorithm for Solving Three-Dimensional Chaotic Equations. *Mathematics*, *10*(21), 4147.https://doi.org/10.3390/math10214147.

[17] Lv S X, Zeng Y R, Wang L. An Effctive Fruit Fly Optimization Algotithm with Hybrid Information Exchange and Its Applications[J]. International Journal of Machine Learning and Cybernetics (S1868-8071), 2018, 9(10): 1623-1648.

[18] Lu, H., Azimi, M., & Iseley, T. (2019). Short-term load forecasting of urban gas using a hybrid model based on improved fruit fly optimization algorithm and support vector

machine. *Energy Reports*, *5*, 666-677. https://doi.org/10.1016/j.egyr.2019.06.003
[19]Li Shaobo, Zhao Hui, Zhao Chenglong, et al. Review of Fruit Fly Optimization Algorithms[J]. Journal of Science Technology and Engineering, 2018, 18(1): 163-171.
[20] Jiang, F., Zhang, W., & Peng, Z. (2022). Multivariate adaptive step fruit fly optimization algorithm optimized generalized regression neural network for short-term power load 10.1109/ICOSEC49089.2020.9215232
forecasting. *Frontiers in Environmental Science*, 292.
https://doi.org/10.3389/fenvs.2022.873939
[21] Abrar, I., Ayub, Z., Masoodi, F., & Bamhdi, A. M. (2020, September). A machine learning approach for intrusion detection system on NSL-KDD dataset. In *2020 international conference on smart electronics and communication (ICOSEC)* (pp. 919-924). IEEE. https://doi.org/