IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received December 25, 2020, accepted January 7, 2021, date of publication January 12, 2021, date of current version January 28, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3051074

Deep Belief Network Integrating Improved Kernel-Based Extreme Learning Machine for Network Intrusion Detection

ZHENDONG WANG, (Member, IEEE), YONG ZENG, YAODI LIU, AND DAHAI LI
College of Information Engineering, Jiangxi University of Science and Technology, Ganzhou 341000, China

Corresponding author: Yong Zeng (zooyoon@163.com)

This work was supported in part by the National Natural Science Foundation of China under Grant 62062037 and Grant 61763017, in part by the Natural Science Foundation of Jiangxi Province under Grant 20181BBE58018, and in part by the Innovation Designated Fund for Graduate Student of Jiangxi Province under Grant YC2019-S298.

ABSTRACT Deep learning has become a research hotspot in the field of network intrusion detection. In order to further improve the detection accuracy and performance, we proposed an intrusion detection model based on improved deep belief network (DBN). Traditional neural network training methods, like Back Propagation (BP), start to train a model with preset parameters such as the randomly initialized weights and thresholds, which may bring some issues, e.g., attracting the model to the local optimal solutions, or requiring a long training period. We use the Kernel-based Extreme Learning Machine (KELM) with the supervised learning ability to replace the BP algorithm in DBN in a bid to ameliorate the situation. Considering the problem of poor classification performance usually caused by randomly initializing kernel parameters with KELM, an enhanced grey wolf optimizer (EGWO) is designed to optimize the parameters of KELM. In order to improve the search ability and optimization ability of the traditional grey wolf optimizer algorithm, a novel optimization strategy combining the inner and outer hunting is introduced. Experiments on KDDCup99, NSL-KDD, UNSW-NB15 and CICIDS2017 datasets show that the proposed DBN-EGWO-KELM algorithm has greater advantages in terms of its accuracy, precision, true positive rate, false positive rate and other evaluation indices compared with BP, RBF, SVM, KELM, LIBSVM, CNN, DBN-KELM and other intrusion detection models, and can effectively meet the requirements of intrusion detection of complex networks.

INDEX TERMS Intrusion detection, deep belief network, kernel-based extreme learning machine, grey wolf optimizer.

I. INTRODUCTION

With the rapid development of network technologies such as 5G [1], cloud computing [2], and the Internet of Things [3], the massive amount of data generated by the network has brought huge difficulties and challenges to network security, a research topic which has attracted more and more attention. Intrusion Detection (ID) [4], a process of marking and identifying intrusions to a network, is a key technique which mainly includes two functions: 1) Analyze the existing network data, record the information characteristics of the existing attack data, and then match it with the data in the host or network. This is a process referred to as misuse detection. 2) Establish a connection between the network data and the normal behavior trajectory characteristics in the sample database. Any deviations from the behavior characteristics are regarded as intrusion behaviors. This is a process referred to as anomaly detection. Misuse detection can record the behavioral characteristics of known cyber attacks, with a low false alarm rate, but it lacks learning ability. Therefore, the matching database must be constantly updated to adapt to the changing environment. In addition, the misuse detection usually has a poor detection effect for new attacks. On the other hand, anomaly detection can effectively detect unknown attacks, but its false alarm rate is high.

The-state-of-the-art of intrusion detection technology mainly focuses on three aspects.

1) Intrusion detection based on data mining. Salo *et al.* [5] identified 19 independent data mining technologies for

The associate editor coordinating the review of this manuscript and approving it for publication was Pedro R. M. Inácio.

intrusion detection based on custom methods, analyzed and compared on their detection performance.

2) Intrusion detection based on machine learning. Wang *et al.* [6] designed a cloud intrusion detection system combined with stacked compression autoencoder and support vector machine. Experimental results show that the system has stronger detection capabilities. Gao *et al.* [7] designed an integrated intrusion detection algorithm with adaptive ability by using decision tree, random forest, neighbor algorithm (KNN) and other basic classifiers, which effectively improved the detection accuracy.
3) Intrusion detection based on neural network. Ahmad *et al.* [8] introduced the Extreme Learning Machine (ELM) to the field of intrusion detection, which can reduce the false alarm rate and improve the detection rate. Naik *et al.* [9] used the teaching-learning metaheuristic optimization algorithm to optimize the parameters of the neural network. The optimized neural network has good results in indicators such as execution efficiency and classification accuracy.

In the intrusion detection system, the network traffic is first identified, and abnormal traffic is blocked. The network attack type is then identified, and the characteristic database of the attack type is continuously improved, thereby improving the system's own defense. In this regard, network anomaly detection can be attributed to binary class classification and multi-classification problems. In recent years, data mining, machine learning, and neural networks have been used by researchers in network anomaly detection and have achieved positive results. However, data mining and traditional machine learning methods rely heavily on the feature extraction and selection of data, which in many cases cannot achieve sufficiently good performance. Therefore, the data processing techniques for, e.g., the accuracy and precision of classification, need to be improved significantly. Hinton [10] successfully reduced and classified MNIST dataset using a deep learning model in 2006, and found that a network model with a deep neural network structure can discover more essential data features in the dataset. Meanwhile, deep learning made remarkable progress in many areas, e.g., control [11], natural language processing [12] and emotional analysis [13], proving the great potential of the framework in the field of data classification. In this regard, some scholars proposed applying deep learning to the field of network security. Khan *et al.* [14] applied deep neural network to network intrusion detection and designed a two-stage deep learning model. The model first uses probability scores to classify network traffic as normal or abnormal, and then uses the scores as an additional feature to detect normal traffic and other attack categories in the decision-making stage. Although the model is able to obtain useful feature representations from unlabeled data to improve the detection accuracy, the disadvantage is that the computing time increases with the size of the dataset. The model proposed in this paper is suitable for large-scale datasets and can process data efficiently in batches. Nie *et al.* [15] established a deep learning model based on convolutional neural network for Internet of Things (IoT) security issues, and designed a data-driven intrusion detection system. This method has higher detection rate and lower false alarm rate, but the learning rate is slow. Su *et al.* [16] proposed a traffic anomaly detection model combined the attention mechanism and the bidirectional long short-term memory network (BLSTM). The model can quickly obtain the key characteristics of network traffic and improve the ability to detect abnormal behaviors, but the generalization ability of the model needs to be improved. While the model proposed in this article does not depend on specific data, it is applicable to different datasets and has strong generalization capabilities. Zhu *et al.* [17] designed a multi-task LSTM neural network intrusion detection system based on the vulnerability of the Internet of Vehicles, and detected abnormal behavior from two dimensions, i.e., the time and the data. This model improved the real-time performance of detection. However, the proportion of the two dimensions is different in different time periods. If the dimensional weight allocation mechanism is not ideal, the detection effect of the intrusion detection system will be adversely affected. Alluri *et al.* [18] proposed a modified binary grey wolf optimisation (MBGWO) feature selection algorithm, which uses support vector machine (SVM) to classify the dataset NSL-KDD. Although the accuracy of intrusion detection is greatly improved, it only verifies the effectiveness of the algorithm on one dataset. It has not been verified by experiments on multiple or newer datasets. However, this paper conducts experiments on four datasets, including large datasets, datasets with many unknown attacks, datasets with many types of attacks, and newer datasets to verify the effectiveness and performance of the model proposed in this article. Riyaz *et al.* [19] proposed a new feature selection algorithm, namely conditional random field and linear correlation coefficient-based feature selection (CRF-LCFS) algorithm, to select the most significant features and classify them using the existing convolutional neural network (CNN). The classification process uses the convolutional layer of CNN to generate feature maps, and the pooling layer to reduce the feature map size, hence shorten the processing time at the expense of reducing the detection accuracy. The model proposed in this paper has no feature map conversion process and uses DBN to reduce the dimensionality of high-dimensional data features, and the corresponding time cost is greatly reduced.

In summary, there are three limitations in the aforementioned research outcomes. Firstly, intrusion detection systems must detect and feedback network traffic in real time, while reducing latency and improving detection efficiency. Secondly, the accuracy and generalization of intrusion detection have much room for improvement. A good model can detect more types of attacks and improve the performance of the intrusion detection system. Incorrectly classified attack data will affect the establishment of intrusion detection models. Thirdly, most of the aforementioned studies are applicable

to the situation where the type of network traffic is known. In the face of unknown attacks, traditional classifiers often cause misjudgments, leading to the degradation or even deterioration of the performance of intrusion detection systems. At this time, the learning ability of the intrusion detection system becomes particularly important. It must have the ability to recognize unknown attacks. In order to solve the above problems, in view of the advantages of deep neural networks in the field of intrusion detection, this paper proposes an intrusion detection model, namely deep belief network based on enhanced grey wolf optimizer and improved kernel based extreme learning machine (DBN-EGWO-KELM). The DBN model is improved to reduce the dimensionality of high-dimensional data features, and then the KELM algorithm is used to replace the traditional BP algorithm for supervised classification, finally the purpose of enhancing the data classification performance of DBN is achieved. In order to improve the generalization ability of the KELM classifier for different datasets, we use the enhanced grey wolf optimizer to optimize the E, s and other parameters of KELM. Experimental results on different datasets show that the DBN-EGWO-KELM model can effectively shorten the training and detection time, and significantly improve the classification accuracy, precision, and true positive rate.

II. RELATED WORK

The security issues of network-based intrusion detection systems (NIDS) coexist since the birth of computer science. Recently, researchers and experts have focused on applying machine learning and neural network-based solutions to NIDS (cf. [20] for a survey). This section mainly discusses network-based intrusion detection systems and our solutions.

A. NETWORK-BASED INTRUSION DETECTION SYSTEMS (NIDS)

Self-learning system is one of the effective methods to deal with current network attacks. It uses the supervised, semi-supervised and unsupervised mechanisms of machine learning, and uses a large number of normal and attacking network events to learn patterns of various normal and malicious activities. However, existing solutions based on machine learning have a high false positive rate and high computational cost. This is because the machine learning classifier simply learns the characteristics of TCP/IP locally. Deep learning is a complex machine learning subnet. It learns feature representations and order relationships by passing TCP/IP information on multiple hidden layers. Deep learning has achieved remarkable results in long-standing artificial intelligence tasks in the fields of image processing [21], speech recognition [22], and natural language processing [23]. Meanwhile, these capabilities have been transformed into various network security tasks, such as intrusion detection, android malware classification, traffic analysis, network traffic prediction, ransomware detection, encrypted text classification, malicious URL detection, anomaly detection, and malicious domain detection [24]. The focus of this work is to analyze the performance of various classic machine learning and deep neural networks (DNNs) on feature extraction from the network-based intrusion datasets, and the effectiveness of NIDS.

A standard benchmark dataset KDDCup99 is used to improve the efficiency of intrusion detection. KDDCup99 was used in the Third International Knowledge Discovery and Data Mining Tools Competition [25], the dataset was created in 1998 in the DARPA Intrusion Detection (ID) Evaluation Network. The purpose of the competition is to create a predictive model that divides network connections into two categories: normal connections and attack connections. Attacks are divided into denial of service (DoS), detection (Probe), remote to local (R2L), user to root (U2R) and other categories. In the KDDCup99 competition, the mining audit data for automated models for ID (MADAMID) was used as feature construction framework. MADAMID outputs 41 characteristics: the first 9 characteristics are basic characteristics of the package, 10-22 are content characteristics, 23-31 are traffic characteristics, and 32-41 are host-based characteristics. The choices of available datasets are: (1) Complete dataset and (2) complementary 10% data. The Competition task has since remained as a baseline job. After this competition, many machine learning solutions have been discovered, most published results use only 10% of training and test data, and very few custom datasets are used. Recently, a comprehensive literature survey of ID-based machine learning was conducted using the KDDCup99 dataset. In [26], the performance of the ID model based on shared nearest neighbor (SNN) is studied, and it is reported as the best detection rate algorithm. By reducing the dataset, they were able to conclude that SNN performed well in the K-means of the U2R attack category. However, their work failed to show results on the entire test dataset. In [27], the naive Bayesian network with root nodes is used to represent connected classes, and leaf nodes are used to represent connected features. Subsequently, a genetic algorithm is proposed based on NIDS [28], which can model time and space information to identify complex abnormal behaviors. In [29], the integrated learning technology is reviewed, and in [30], the swarm intelligence technology of ant colony optimization, ant colony clustering and particle swarm optimization is studied. A comparative study in such research works shows that the descriptive statistics was predominantly used.

In general, a comprehensive literature review shows that few studies use modern deep learning methods for NIDS. The commonly used experimental analysis benchmark datasets are KDDCup99 and NSL-KDD [31]–[33].

B. THE WORK FLOW

The network intrusion detection model we proposed is mainly composed of three stages. The first stage is to process the dataset, the second stage is the establishment of the classification model, and the third stage is the intrusion recognition process. By combining the three stages, we obtain a specific process which is able to improve the structure of the DBN

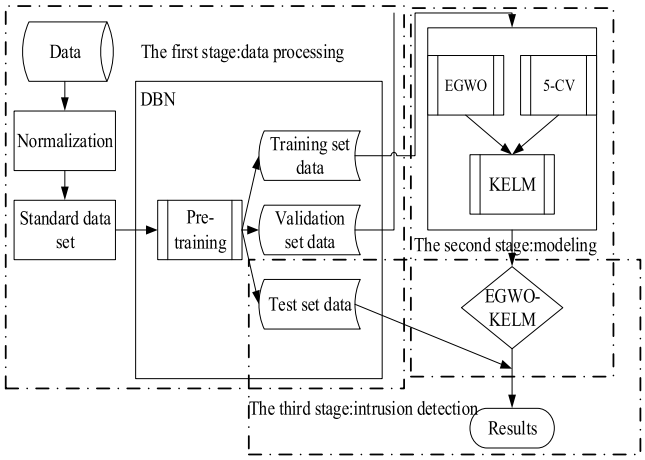


FIGURE 1. Overall framework.

model, while retaining the ability of DBN to reduce the dimensionality of high-dimensional data features, using the KELM algorithm to replace the traditional BP algorithm for supervised classification to enhance the data classification performance of DBN. The enhanced Grey wolf optimizer is used to optimize the parameters, such as E and s of KELM, to improve the generalization ability of KELM classifier for different datasets. The overall framework is shown in Fig 1.

III. NETWORK MODEL

In this section, we will focus on the network model proposed in this paper by introducing the DBN and KELM first. According to the structure and characteristics of the network model, some improvement ideas and methods are presented in detail.

A. DEEP BELIEF NETWORK

As Hinton proposed in 2006, the deployment of DBN is composed of multiple Restricted Boltzmann Machines (RBM) stacked [34]. The network first uses the Contrastive Divergence (CD) algorithm for unsupervised training of the stacked RBM, and then uses the Back Propagation (BP) algorithm to fine-tune the node parameters in the entire DBN network. The structure is shown in Fig. 2. DBN training mainly includes two stages: pre-training and fine-tuning. The pre-training stage uses each layer of RBM to perform unsupervised training on unlabeled sample data, and at the same time uses the CD algorithm to tune each layer of RBM parameters. After training at each RBM layer is over, the parameters obtained from the training of a RBM layer are sent to the next RBM layer for training, until all RBM layers have completed training. After the pre-training, DBN calculates the network errors of each layer through the BP algorithm, and adjusts the parameters of each layer node through back propagation, so as to realize the global fine-tuning of the node weights of the entire DBN network. Compared with the traditional neural network, the weights of each layer of DBN are trained before the test, instead of random initialization. Therefore, the model can overcome the shortcomings of the traditional neural network such as being easy to fall into the local optimum, and a long training period. However, because the initial parameters of the BP algorithm are randomly generated, the amount of iteration calculation in the network is large. This brings defects such as random parameters when the BP algorithm fine-tunes the DBN node parameters. This process not only increases the training time, but also leads to poor stability of the network.

1) Restricted Boltzmann Machines (RBM): The undirected graph model includes the visual layer and the hidden layer. Each layer has several nodes. There is no connection between nodes in the same layer, while the nodes between the visible layer and the hidden layer are fully connected. The structure is shown in Fig. 3. RBM is an energy-based model, and energy function is defined by the visible layer $v = (v_i)_n$ and the hidden layer $h = (h_j)_m$:

$$E_\theta(v, h) = -\sum_{i=0}^{n_v} a_i v_i - \sum_{j=0}^{n_h} b_j h_j - \sum_{i=0}^{n_v}\sum_{j=0}^{n_h} v_i w_{ij} h_j \quad (1)$$

a_i is the bias of the i-th neuron in the visible layer and b_j is the bias of the j-th neuron in the hidden layer;$\theta = [w = (w_{ij})_{n\times m}, a = (a_i)_n, b = (b_j)_m]$ is the parameters of the RBM model; w_{ij} is the connection weight between the visible layer v_i and the hidden layer h_j; n_v represents the number of visible layers, and n_h represents the number of hidden layers.

The joint probability distribution of state (V, H) can be obtained from the energy function by Eq. (1):

$$P_\theta(v, h) = \frac{1}{Z_\theta}\exp(-E_\theta(v, h)) \quad (2)$$

Among them, Z_θ is the normalization factor by Eq. (2):

$$Z_\theta = \sum_v \sum_h \exp(-E_\theta(v, h)) \quad (3)$$

For RBM, all neuron states in the hidden layer are independent of each other. When the state v of the neuron on the visible layer is given, the probability that the j-th neuron h_j in the hidden layer is activated (with probability 1) is

$$P_\theta(h_j = 1 \mid v) = sigmoid\left(b_j + \sum_{i=0}^{n_v} w_{ij} v_i\right) \quad (4)$$

When the state h of the neuron on the hidden layer is given, the probability that the i-th neuron v_i in the visible layer is activated (with probability 1) is

$$P_\theta(v_i = 1 \mid h) = sigmoid\left(a_i + \sum_{j=0}^{n_v} w_{ji} h_j\right) \quad (5)$$

In Eq. (5), $sigmoid(x) = (1 + \exp(-x))^{-1}$ is the activation function, where x is in the interval (0,1).

For an RBM model with a given number of neurons in the visible layer and hidden layer, it is necessary to train the RBM to determine the parameter θ, to ensure that the RBM model controlled by the parameter θ fits the given training data as much as possible. Due to the existence of the normalization factor Z_θ, it is difficult to use the naive method to calculate $P_\theta(v, h)$,while the CD algorithm can be used to quickly train the RBM model in an unsupervised mode with fewer samples, to achieve the purpose of obtaining the optimal solution of the parameter θ.

According to CD algorithm, the parameter update method is as follows:

$$\begin{cases} \Delta w_{ij} = \varepsilon\left(\left\langle v_i^0 h_j^0\right\rangle_{data} - \left\langle v_i^1 h_j^1\right\rangle_{recon}\right) \\ \Delta a_i = \varepsilon\left(\left\langle v_i^0\right\rangle_{data} - \left\langle v_i^1\right\rangle_{recon}\right) \\ \Delta b_j = \varepsilon\left(\left\langle h_j^0\right\rangle_{data} - \left\langle h_j^1\right\rangle_{recon}\right) \end{cases} \tag{6}$$

Among them: ε is the learning rate of the algorithm, $\langle\cdot\rangle_{data}$ is the mathematical expectation of the training dataset, and $\langle\cdot\rangle_{recon}$ is the target mathematical expectation calculated by the CD algorithm.

2) Deep belief neural network pre-training: The DBN model is formed by stacking multiple RBMs, including an input layer, multiple hidden layers, and an output layer composed of a BP neural network. The network structure is shown in Fig. 4. In DBN, every two adjacent layers form an RBM, and the input layer of DBN is the first layer RBM_0, the hidden layer neuron h_0 of this layer is taken as the visible layer neuron v_1 of the next layer RBM_1, and so on. In the last visible layer RBM_k, neuron v_k is the hidden layer neuron h_{k-1} of the layer RBM_{k-1}. The top layer of the DBN is the BP algorithm, which maps the data features extracted from the original data by the RBM to the categories to be classified.

DBN pre-training process: Firstly, the unlabeled sample data is used to train the stacked RBM layer by layer. In this process, the CD algorithm is used to obtain the parameter θ_0 of the first layer RBM_0. Secondly, input the value of θ_0 into the next layer to continue training RBM_1, until the parameter θ_1 is obtained, and so on, to obtain the bias and weight of the entire DBN.

3) Supervised fine-tuning based on BP algorithm: Since the training between each RBM is completed independently, it can only ensure that the node parameter values obtained by the training are optimal within the respective RBM. Therefore, after the DBN pre-training is completed, the parameters in each RBM need to be fine-tuned. The sample dataset $\{a, b\}$ is given, the relationship between input and output is:

$$\widehat{b_i} = f(a_i, \theta) \tag{7}$$

Among them: $\widehat{b_i}$ is the i-th sample of the DBN mapping, f is a non-linear function, and a_i is the i-th sample in the training sample dataset $a = [x_1, x_2, \ldots, x_n]$.

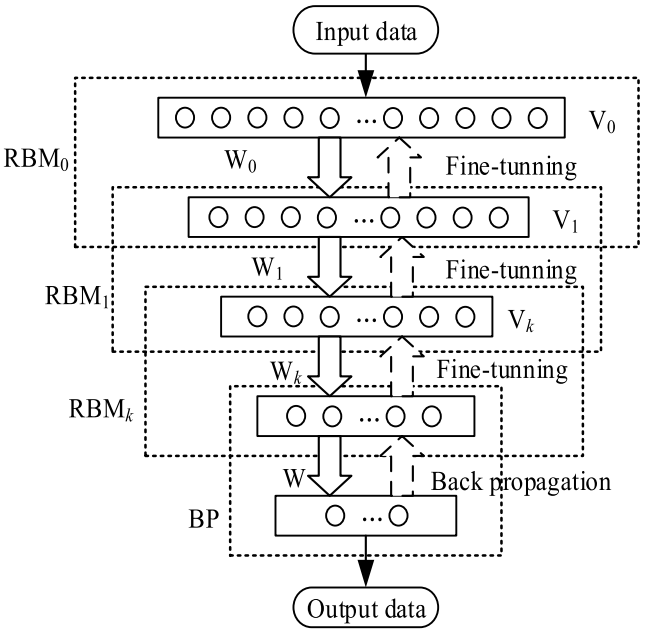


FIGURE 2. Deep belief network framework.

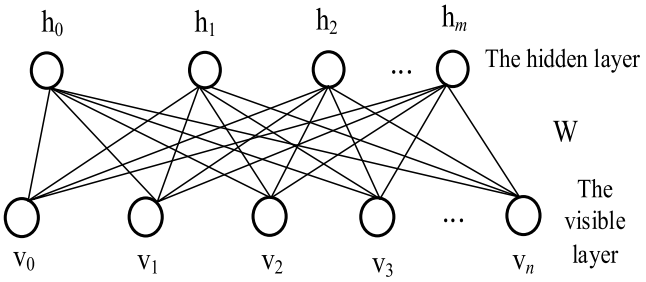


FIGURE 3. RBM structure.

The error loss function is represented by the average cross entropy between the predicted value and the actual value:

$$j(\theta) = \frac{1}{N}\sum_{i=1}^{N} b_i \ln\left(\widehat{b_i}\right) \tag{8}$$

b_i in Eq. (8) is the actual value of the i-th sample in the sample dataset.

In DBN, the supervised fine-tuning based on the BP algorithm generally adopts the gradient descent method. This type of method is difficult to obtain a learning rate, and it is easy to fall into the local optimum and the amount of iteration is very large [35], hence a long delay. According to literature [37]–[41], the kernel-based extreme learning machine can effectively solve the problems of BP neural network, and has a strong classification ability. Therefore, this paper uses the kernel-based extreme learning machine instead of BP algorithm to achieve supervised classification. Later, we will explain in detail the benefit of using the kernel-based extreme learning machine in place of the BP algorithm.

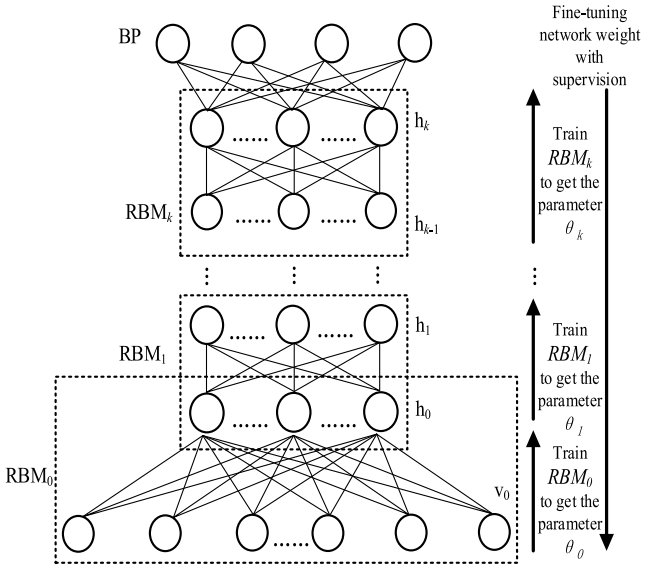


FIGURE 4. Structure of deep belief network.

B. KERNEL-BASED EXTREME LEARNING MACHINE

Extreme Learning Machine (ELM) was proposed by Huang [36] through the theory of generalized inverse matrix. Compared with traditional neural networks, ELM improves the learning speed while maintaining good generalization capabilities of the network, and has strong nonlinear fitting capabilities, which can effectively reduce the amount of calculation as well as the search space. Based on the above advantages, ELM has been applied in many fields, such as information collection [37], big data application [38], logo recognition [39], language recognition [40]. Kernel-based Extreme Learning Machine [41] combines the kernel function on the basis of ELM. The linearly inseparable high-dimensional information is projected to the high-dimensional feature space through nonlinear mapping to achieve linear separability, in order to achieve the purpose of improving the accuracy of classification. However, due to the combination of the kernel function, KELM is sensitive to parameter settings. To tackle this problem, this paper uses the swarm intelligence optimization algorithm to optimize the KELM parameters to improve the efficiency of parameter tuning. Extensive experiments, and comparisons with other swarm intelligence algorithms, demonstrate that the grey wolf optimizer has outstanding capabilities. This work improves the grey wolf optimizer strategy and enhances its performance. The improved grey wolf optimizer will be described in detail below.

IV. ENHANCED GREY WOLF OPTIMIZER AND KERNEL PARAMETER OPTIMIZATION MODEL

In this section, we will explain the use of the enhanced grey wolf optimizer, and introduce the model of the enhanced grey wolf optimizer to optimize the kernel parameter.

A. ENHANCED GREY WOLF OPTIMIZER

As a meta-heuristic algorithm [42], GWO was first proposed by Mirjalili and others in 2014. The algorithm divides the wolves into four levels, from high to low as α, β, δ and ω. Among them, α, β, and δ are the three levels closest to the prey, that is, the three highest levels in the grey wolf hierarchy in Fig. 5. The remaining grey wolf individuals are ω, which has the lowest level. The higher three levels lead ω to hunt prey. It is worth noting that the social hierarchy of grey wolves is not fixed. With the progress of the encirclement, all grey wolves will be reclassified according to the distance between themselves and their prey, that is to say, they will be re-layered according to the fitness value update result. In the process of re-layering, the position of the head wolf may not be optimal when the grey wolf position vector is updated, so it is difficult to balance the global and local search capabilities, which leads to problems such as the GWO algorithm falling into the local optimum and slower convergence speed in the iterative process. In this regard, this article improves the original grey wolf optimizer. According to reference [43],when $\left|\vec{A}\right| < 1$, the grey wolf population will narrow the search range and perform a fine search in a local area, which is the development capability of the GWO algorithm. when $\left|\vec{A}\right| > 1$, the grey wolf population will expand the search range to find a better candidate solution, which is the global exploration capability of the GWO algorithm. In order to improve the convergence performance of the algorithm, we designed an enhanced grey wolf optimizer (EGWO) by strengthening the development capabilities of GWO. In EGWO, we divide the grey wolf population strategically. Half of the grey wolves are responsible for the inner hunting, while the other half are responsible for the outer hunting. The inner hunting is used to enhance the grey wolf's ability to attack prey, i.e., the algorithm development ability. Outer hunting is used to maintain the global exploration capabilities of GWO. In order to further improve the optimization accuracy of the algorithm, the Tent mapping [43] is used to enrich the diversity of the initial population. The schematic diagram of the inner and outer hunting is shown in Fig. 6.

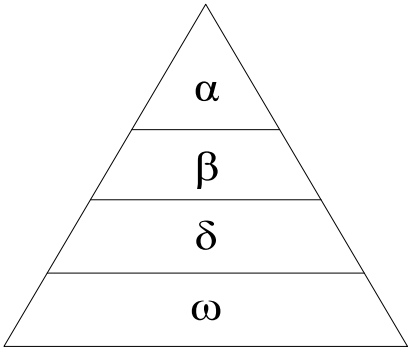


FIGURE 5. Grey wolf social hierarchy pyramid.

1) The outer hunting: The global search process means that the grey wolf population needs to explore a wider search area in order to find the global optimal solution. Therefore, the grey wolves carrying out the outer encirclement order approach the prey from all directions and

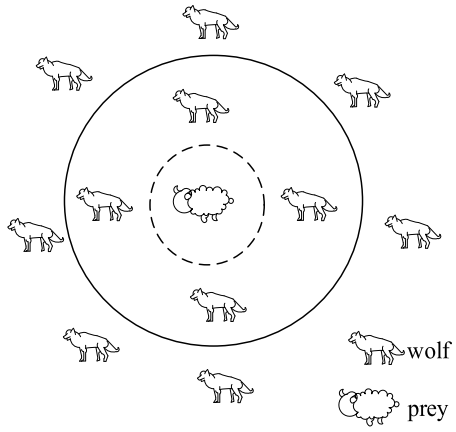


FIGURE 6. Inner and outer hunting of grey wolf.

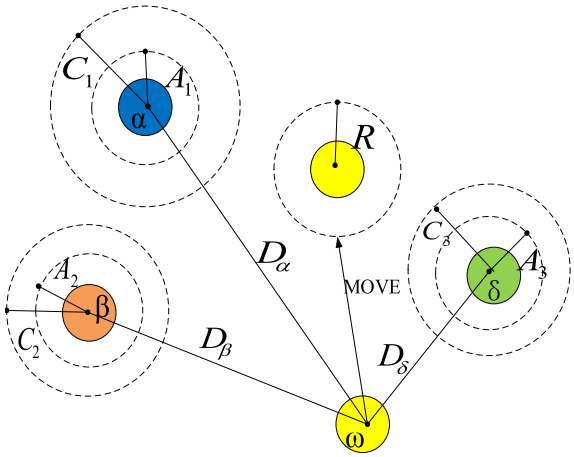


FIGURE 7. Grey wolf location update.

from far to near. In order to avoid the algorithm falling into the local optimum, the grey wolves performing the outer encirclement must obey the commands of α, β and δ. Fig. 7 is a schematic diagram of the grey wolf update location, the location of α, β and δ wolves are used to simulate the approximate position of prey. When ω receives the command to kill, it moves closer to the prey to update its position. The distance between ω and α, β and δ is expressed by the following formula:

$$\begin{cases} \vec{D}_\alpha = \left|\vec{C}_1 \cdot \vec{X}_\alpha(t) - \vec{X}_\omega(t)\right| \\ \vec{D}_\beta = \left|\vec{C}_2 \cdot \vec{X}_\beta(t) - \vec{X}_\omega(t)\right| \\ \vec{D}_\delta = \left|\vec{C}_3 \cdot \vec{X}_\delta(t) - \vec{X}_\omega(t)\right| \end{cases} \tag{9}$$

$$\begin{cases} \vec{C}_1 = 2 \cdot \vec{r}_1 \\ \vec{C}_2 = 2 \cdot \vec{r}_2 \\ \vec{C}_3 = 2 \cdot \vec{r}_3 \end{cases} \tag{10}$$

Among them,$\vec{D}_\alpha$ represents the distance between α and ω, and so on for $\vec{D}_\beta$, $\vec{D}_\delta$. $\vec{X}_\alpha(t)$ is the position of α in the t-th iteration, and so on for $\vec{X}_\beta(t)$, $\vec{X}_\omega(t)$. $\vec{C}_1$ is the azimuth variable when ω moves to α, and so on for $\vec{C}_2$, $\vec{C}_3$. $\vec{r}_1$ is a random number of [0,1].

After ω knows the distance to α, β and δ, it will approach each with a certain step length, and its position update can be expressed as follows:

$$\begin{cases} \vec{X}_1 = \vec{X}_\alpha - \vec{A}_1 \cdot \vec{D}_\alpha \\ \vec{X}_2 = \vec{X}_\beta - \vec{A}_2 \cdot \vec{D}_\beta \\ \vec{X}_3 = \vec{X}_\delta - \vec{A}_3 \cdot \vec{D}_\delta \end{cases} \tag{11}$$

$$\begin{cases} \vec{A}_1 = 2a \cdot \vec{r}_4 - a \\ \vec{A}_2 = 2a \cdot \vec{r}_5 - a \\ \vec{A}_3 = 2a \cdot \vec{r}_6 - a \end{cases} \tag{12}$$

$$a = 2 - 2\left(\frac{t}{T}\right) \tag{13}$$

$$\vec{X}(t+1) = \frac{\vec{X}_1 + \vec{X}_2 + \vec{X}_3}{3} \tag{14}$$

$\vec{X}_1$ represents the updated position of ω if only α is the prey, and so on for $\vec{X}_2$, $\vec{X}_3$. t is the current number of iterations, T is the maximum number of iterations, a is the convergence factor, and its role is to reduce the distance between the grey wolf and its prey as the iteration progresses. $\vec{A}_1$ is the step vector of ω moving in the direction of α, and so on for $\vec{A}_2$, $\vec{A}_3$. Under the joint leadership of α, β and δ, its position can be expressed as the arithmetic mean of$\vec{X}_1$, $\vec{X}_2$ and $\vec{X}_3$. The population location update strategy of outer encirclement is carried out by formula (9-14).

2) The inner hunting: The local development ability uses the existing information to influence other search, and then agents to perform fine searches in certain search space, which greatly affects the convergence performance of the algorithm. Therefore, EGWO focuses on improving the development capabilities of the algorithm. In the EGWO algorithm, the change of α has positive impact on the hunting strategy by making the algorithm more intelligent in the hunting process. A more detailed encircle order is arranged for the grey wolf group. The distance between the grey wolf group and the prey that executed the inner hunting command is short, just follow the order of α. The position of α is used to simulate the position of the prey. Therefore, the grey wolf group that executes the inner-layer encircle only needs to move around α. Its position update strategy is similar to GWO, though still different. The grey wolf location update strategy is shown in formula (15):

$$\begin{cases} \vec{D'}_\alpha = \left|\vec{C}_4 \cdot \vec{X}_\alpha(t) - \vec{X}_\alpha(t)\right| \\ \vec{C}_4 = 2 \cdot \vec{r}_7 \\ \vec{X'}_l = \vec{X}_\alpha - \vec{A}_4 \cdot \vec{D'}_\alpha \\ \vec{A}_4 = 2a \cdot \vec{r}_8 - a \\ \vec{X}(t+1) = \vec{X'}_l \end{cases} \tag{15}$$

TABLE 1. Enhanced grey wolf optimizer.

Algorithm : Enhanced Grey Wolf Optimizer

Input: Population size N, maximum number of iterations
Output: Grey wolf X_α

Initialize the grey wolf population vector **W** by

$$x_{t+1} = \begin{cases} \dfrac{x_t}{0.6}, & 0 \le x_t \le 0.6 \\ \dfrac{1 - x_t}{0.4}, & 0.6 < x_t \le 1 \end{cases}$$

$t = 1$
while $t < T$ **do**
  Initialize/update a, A and C
  Calculate the fitness value of **W** by

$$fitness = fit(X)$$

  The best grey wolf in **W** as X_α
  The second best grey wolf in **W** as X_β
  The third best grey wolf in **W** as X_δ
  for $i = 1$ to $N/2$ **do**
    Calculate D_α, D_β, D_δ, X_1, X_2 and X_3 by Eqs. (9-13)
    Update the position of ω by Eq. (14)
  end
  for $i = N/2+1$ to N **do**
    Calculate and by Eq. (15)
    Update the position of ω by Eq. (15)
  end
  $t = t + 1$
end

It can be seen from the above formula that the update strategy is similar to GWO. The difference is that in the inner hunting, we cancel the influence of β and δ on ω, and ω obeys the command of α uniformly. $\vec{D}'_\alpha$ is the distance between ω and α, $\vec{r}_7$ is a random decimal vector within [0,1], $\vec{C}_4$ is the corresponding direction vector, and $\vec{A}_4$ is the corresponding step vector.

3) The EGWO pseudo code: Based on GWO, inner hunting and outer hunting strategies, the pseudo code of EGWO is shown in the following table.

Assume that the maximum number of iterations of the algorithm is T, the size of the population is N, the dimension of the optimization problem is D, and the time complexity of the initialization of the population is O(ND). In GWO, first of all, the fitness value of each grey wolf needs to be calculated and the best three grey wolves are found, with a time complexity of O(N). Later, the location of all grey wolves is updated, and its time complexity is O(N). The total time complexity of each iteration is O($N + N$), so the total time complexity required by GWO is O($T(N + N)+ND$), which is at the level of O(TN). Compared with GWO, EGWO adds an inner-layer encircle mechanism to the location update strategy, and the time complexity of the grey wolf population location update is still O(N). In addition, the composition structure of GWO and EGWO is the same. Therefore, no additional time complexity is added, so the time complexity of EGWO is consistent with GWO, which is O(TN). However, from the experimental results of the benchmark test function, the EGWO algorithm has higher solution accuracy, faster convergence speed and better stability. The details of the experiment are described in the next paragraph.

The performance of the four optimization algorithms of EGWO, GWO, PSO and FPA were tested under three single-peak benchmark functions, Sphere, Schwefel2.22, and Rosenbrock, and three multi-peak benchmark functions Rastrigin, Ackley, and Griewank. The initial value and dimension of the population are both set to 30, the maximum number of iterations is 500, other parameters are the same. Each group of experiments were run for 20 times, and then the average value was taken for comparison. The experimental results are shown in Fig. 8. It can be seen that the EGWO algorithm has the best performance in terms of solution accuracy, convergence speed and stability.

B. KERNEL PARAMETER OPTIMIZATION MODEL

The EGWO algorithm is used to optimize the kernel parameters of KELM (c.f. part B of II), the experiment in part A of III also proved that the EGWO algorithm has good performance in terms of solution accuracy, convergence speed and stability. In this part, we will design a kernel parameter optimization model by combining the two algorithms, namely EGWO-KELM.

The normalization coefficient E and the kernel parameter s are the key to KELM parameter optimization. The KELM classification accuracy rate is recorded as $acc\,(E, s)$, the upper bounds of the two parameters are m and n, and the lower bound is 0. The KELM parameter optimization model can be expressed as:

$$maxacc\,(E, s)\,, E \in (0,\ m]\,, s \in (0,\ n] \tag{16}$$

In order to obtain the relevant parameters of the KELM classifier with the highest classification accuracy on the specified dataset, we use 5-CV [44], [45] to generate random training set, validation set and test set, and then the average accuracy of the five training models is used as the evaluation index of KELM classifier. The cross-validation accuracy rate is shown in the following formula:

$$acc\left(\hat{f}\right) = \frac{1}{N}\sum_{i=1}^{N} d\left(y_i, \hat{f}^{-k(i)}\left(x_i\right)\right) \tag{17}$$

Among them, $d\left(y_i, \hat{f}^{-k(i)}\left(x_i\right)\right)$ is the accuracy of the classification model after verification on the i-th fold, $\hat{f}^{-k(i)}\left(x_i\right)$ is the classification model trained on the dataset after removing the i-th fold of the classifier. The average classification accuracy of each classification model is the cross-validation accuracy.

As mentioned earlier, KELM is sensitive to its own parameter changes, and there are a large number of local extrema. Since the gradient search method is not effective, the EGWO algorithm is used for parameter optimization. We combine

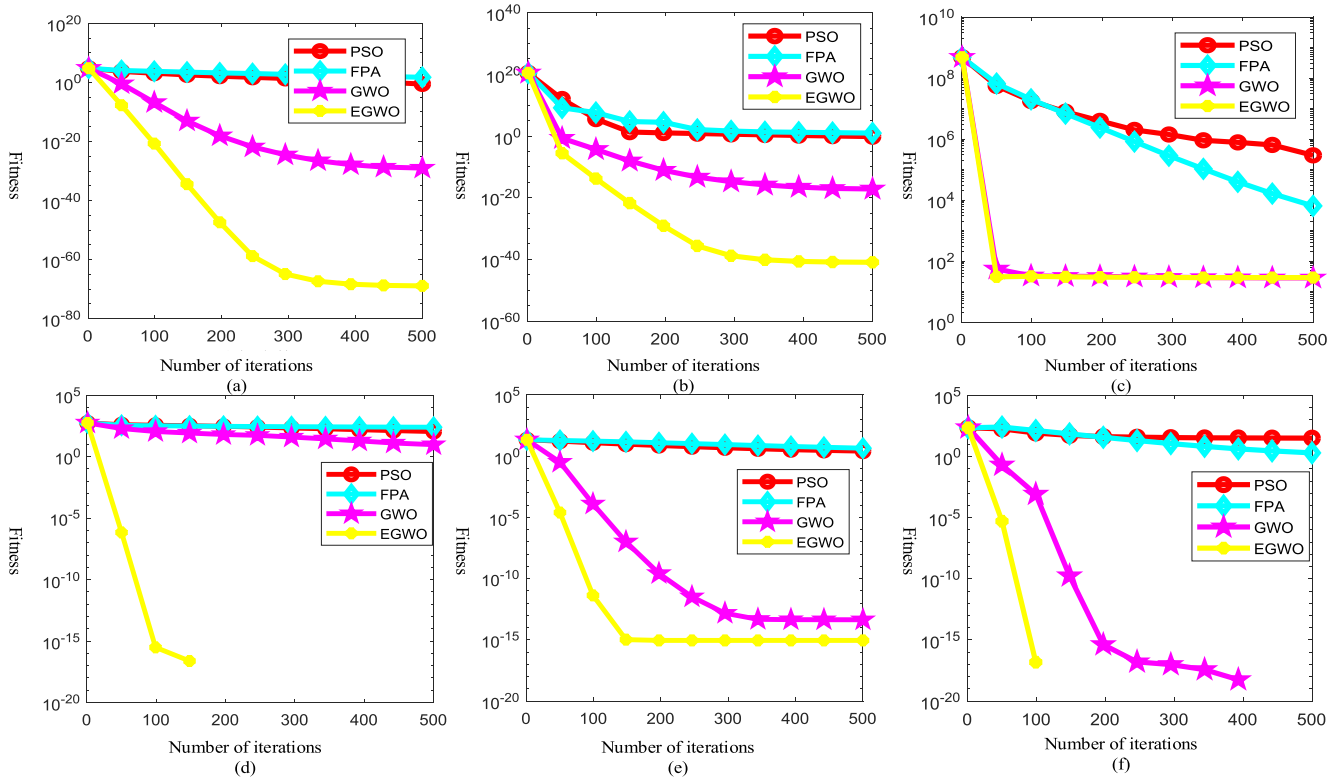


FIGURE 8. Test results graph of algorithm.

5-CV with EGWO algorithm to optimize the parameters of KELM classifier, the steps are as follows.

1) Initialize the wolf pack according to Tent mapping and Eq. (10), (12) and (13), and set the maximum number of iterations T;

2)According to Eq. (9-15), update the location information of the wolf and carry out the inner and outer hunting strategy;

3) Using the 5-CV method, calculate the cross-validation accuracy rate according to Eq. (17) and use this value as the individual fitness to evaluate the classifier parameters;

4) Judge whether the iteration stop condition is satisfied. If not, turn to step 2. Otherwise, output the optimal parameters and the optimal model, and the algorithm ends.

The flowchart of the kernel parameter optimization model is shown in Fig. 9.

In order to verify the classification performance of the above-mentioned EGWO-KELM, a classification comparison experiment with the KELM and SVM methods was done on the UCI Iris, Segment and Diabetes datasets. The experimental results are shown in Table 2.

It can be seen that on the Iris dataset, the classification accuracy of the EGWO-KELM model is as high as 96.12%, and the training time is the shortest, which is 0.326s. On the Segment dataset, the training time of the BP algorithm is 23.39s, while the training time of the EGWO-KELM model is only 1.004s, which is 22.386s shorter than the BP algorithm, and the classification accuracy is still the highest, 95.92%.

TABLE 2. Classification performance comparison of different methods under different datasets.

Dataset	Training set	Test set	Algorithm	Training time/s	Accuracy/%
Iris	100	50	BP	2.031	90.61
			SVM	0.505	95.56
			KELM	0.437	95.57
			EGWO-KELM	0.326	96.12
Segment	1610	700	BP	23.39	89.26
			SVM	7.985	95.79
			KELM	2.368	95.92
			EGWO-KELM	1.004	95.92
Diabetes	644	124	BP	8.658	71.68
			SVM	0.198	79.88
			KELM	0.089	82.25
			EGWO-KELM	0.046	83.56

Similarly, the classification accuracy of the EGWO-KELM model on the Diabetes dataset has reached 83.56%, and the training time is 0.046s, which is the best among the four comparison models. In summary, the EGWO-KELM model has shown superior performance to the other three models on different datasets, in terms of both training time and classification accuracy.

To illustrate the sensitivity of the aforementioned KELM classifier to the normalization coefficient E and the kernel

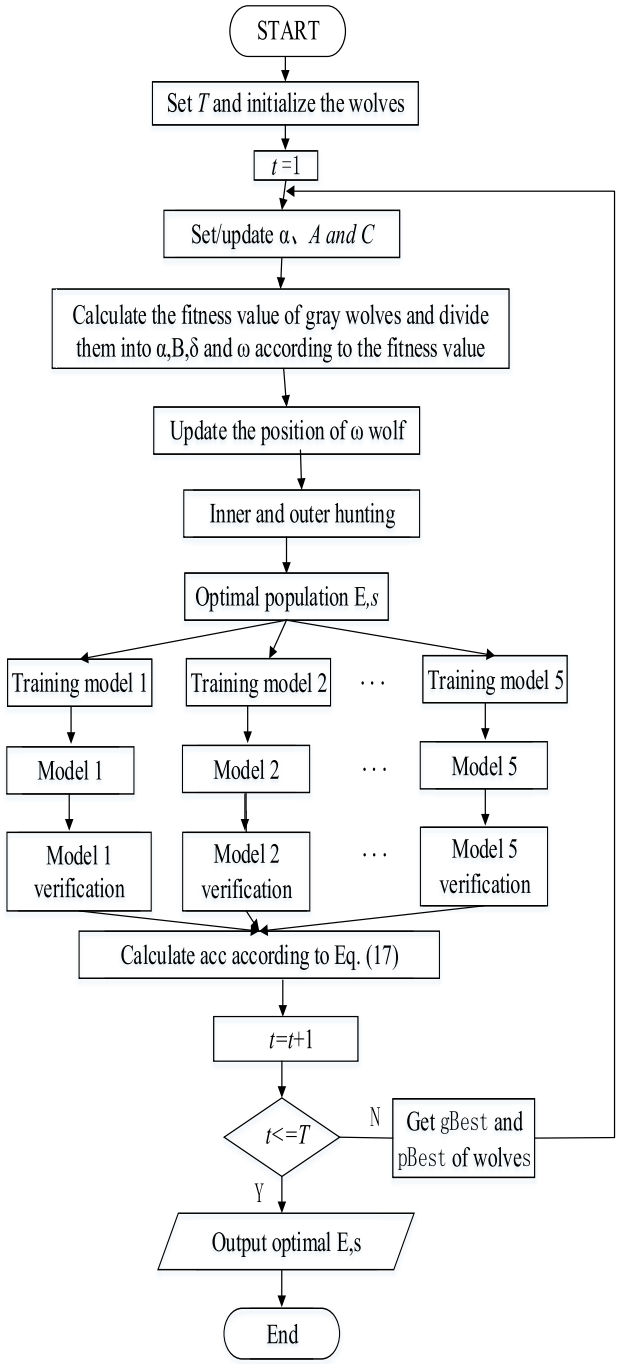


FIGURE 9. Parameter optimization based on 5-CV and EGWO.

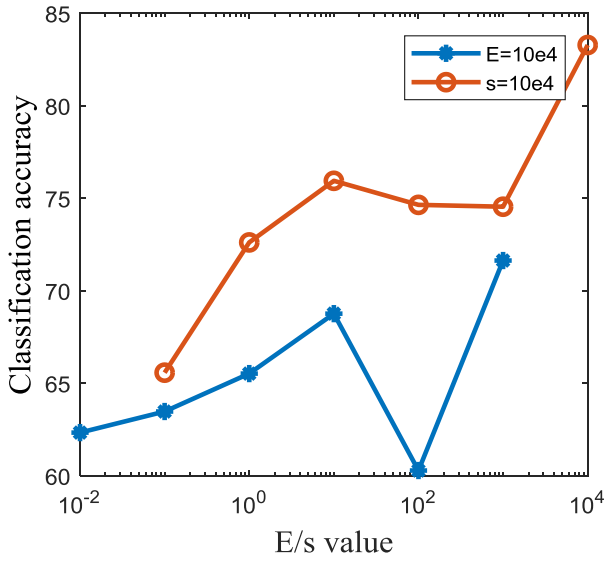


FIGURE 10. Influence of parameter E, s selection on classification accuracy.

parameter *s*, Fig. 10 shows the classification accuracy of the EGWO-KELM classifier under different parameter values on the Diabetes dataset.

It is obvious in Fig.10 that when the parameter $E=$ 10e4 and the value of *s* is 100, the EGWO-KELM model has the lowest classification accuracy on the Diabetes dataset, which is 60.29%. When the value of *s* is 1000, the classification accuracy is the highest, 71.64%; when the parameter $s=$ 10e4 and the value of *E* is 10000, the classification accuracy is the highest, which is 83.29%, compared with the lowest classification accuracy of $E=$ 0.1, the classification accuracy is 17.71% higher. It can be seen that the parameter selection of the EGWO-KELM model has a remarkable impact on the classification accuracy, which affects the classification performance of the classifier significantly. Therefore, we choose to combine 5-CV and EGWO algorithms to optimize KELM classifier parameters, and designed the kernel parameter optimization model EGWO-KELM.

From the classification results of EGWO-KELM on the test sets, we know that EGWO-KELM has superior classification capabilities. Later, we will apply it to DBN to replace the BP algorithm to enhance the classification capabilities of DBN.

V. INTRUSION DETECTION ALGORITHM DBN-EGWO-KELM BASED ON IMPROVED DBN

The enhanced grey wolf optimization algorithm can optimize the deep belief network intrusion detection model of the kernel-based extreme learning machine (DBN-EGWO-KELM). The first part of the model uses DBN feature dimensionality reduction capabilities to extract key features from the standard dataset after data preprocessing. The datasets are divided into training set, validation set and test set, and used as the input into the second part of the EGWO-KELM classification model. This part uses the strategy of combining inner and outer hunting to improve the grey wolf optimizer, which improves the algorithm in terms of having higher solution accuracy, faster convergence speed and better stability. Then the revised algorithm, combined with the 5-CV method, is applied to the parameter optimization of KELM to avoid potential defects caused by KELM random initialization parameters. The optimized EGWO-KELM is used to replace the BP algorithm for network training on the training

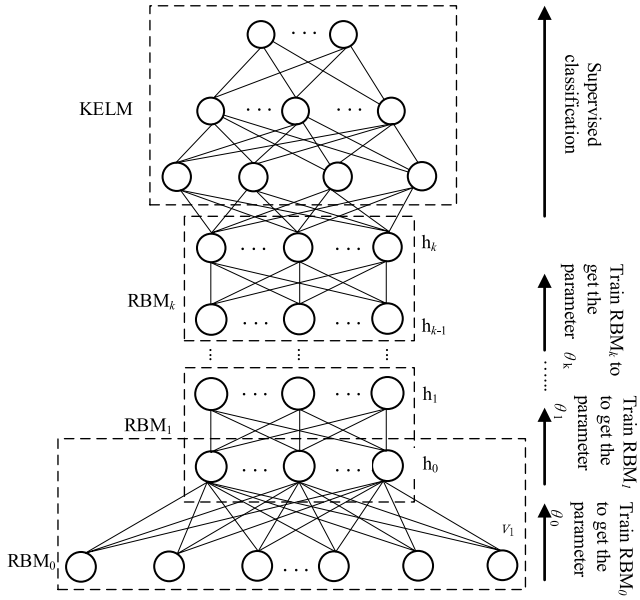


FIGURE 11. DBN-EGWO-KELM structure.

set and validation set, form the EGWO-KELM classification model to classify the test set.

A. CLASSIFICATION MODEL BASED ON DBN-EGWO-KELM

The original DBN model uses the traditional BP algorithm, and the direct use of DBN potentially has the following problems.

1) Random initialization parameters of BP algorithm cause learning to converge to the local optimal solution;

2) BP algorithm training time is long, the amount of iteration and calculation is large, which leads to the slow learning process of DBN;

3) The performance of BP algorithm in classification problem is general, and its feedback network traffic is not strongly in real time.

Given the superior performance of KELM in the classification performance experiment, to solve the above problems, the EGWO-KELM model is used to replace the BP algorithm for supervised classification. Here, EGWO is used to enhance the stability of the KELM classifier, and EGWO-KELM is able to remarkably improve the classification performance and generalization ability of DBN, and effectively improve the accuracy and efficiency of intrusion detection. The structure of the DBN-EGWO-KELM classification model is shown in Fig. 11.

B. DBN-EGWO-KELM INTRUSION DETECTION PROCESS

DBN-EGWO-KELM intrusion detection process includes the following steps. The flowchart is shown in Fig. 12.

1)Data preprocessing. Numericalize the character features in the KDDCup99, NSL-KDD, UNSW-NB15 and CIDIDS2017 datasets, and then normalize the data to obtain numerical data to form a standardized dataset.

2)Define DBN-EGWO-KELM neural network model parameters. According to the results of the parameter search experiment, the network classification performance is the best when iterates 55 times and the number of hidden layers is 80.

3)DBN feature dimensionality reduction. Perform pre-training on the pre-processed dataset, determine the connection weights in the network, and obtain low-dimensional representation data.

4)Data separation. The data after dimensionality reduction is divided into training set, validation set and test set in proportion.

5)Use the parameter optimization model of part B in **III** to optimize the parameters of the KELM classifier. Output the optimal E, s parameters and apply to KELM.

6)Form EGWO-KELM supervised classification model. The training set and validation set are used as the input to the EGWO-KELM classification model for training, and the model is subsequently adjusted

7)Adjust and find the best classification model, and output the best EGWO-KELM classification model.

8) Use test set to test the model, and output results.

VI. SIMULATION EXPERIMENTS AND ANALYSIS

We did four sets of experiments in total. The experimental environment is windows7 64-bit operating system, processor Intel(R)Core(TM)i5-6500 CPU 3.20GHz, installed memory (RAM) 8.00GB.

1) Parameter setting experiment. Use the test function to test the convergence performance of the EGWO algorithm, and search the parameters of the DBN model to determine the number of network iterations and the number of hidden layers;

2) EGWO-KELM classification performance experiment. Using UCI Iris, Segment and Diabetes datasets to do classification comparison experiments to verify the classification performance of EGWO-KELM;

3) Binary classification experiment. Binary classification experiments are carried out on four datasets by different methods, with their classification performance compared with multiple evaluation indicators.

4) Multi-classification experiments. Experimental verification of the classification performance of the DBN-EGWO-KELM model on the KDDCup99, NSL-KDD, UNSW-NB15 and CICIDS2017 datasets.

A. EXPERIMENTAL PARAMETER SETTING

1)The parameters of the enhanced grey wolf optimizer [46] in the classification model DBN-EGWO-KELM are shown in Table 3.

2)DBN network parameter setting. The parameters of the improved DBN classification model are determined by the classification performance analysis experiment and the parameter search experiment. The dataset is divided into three parts for training sample data Train, validation sample data Validation and testing sample data Test. There is no overlap

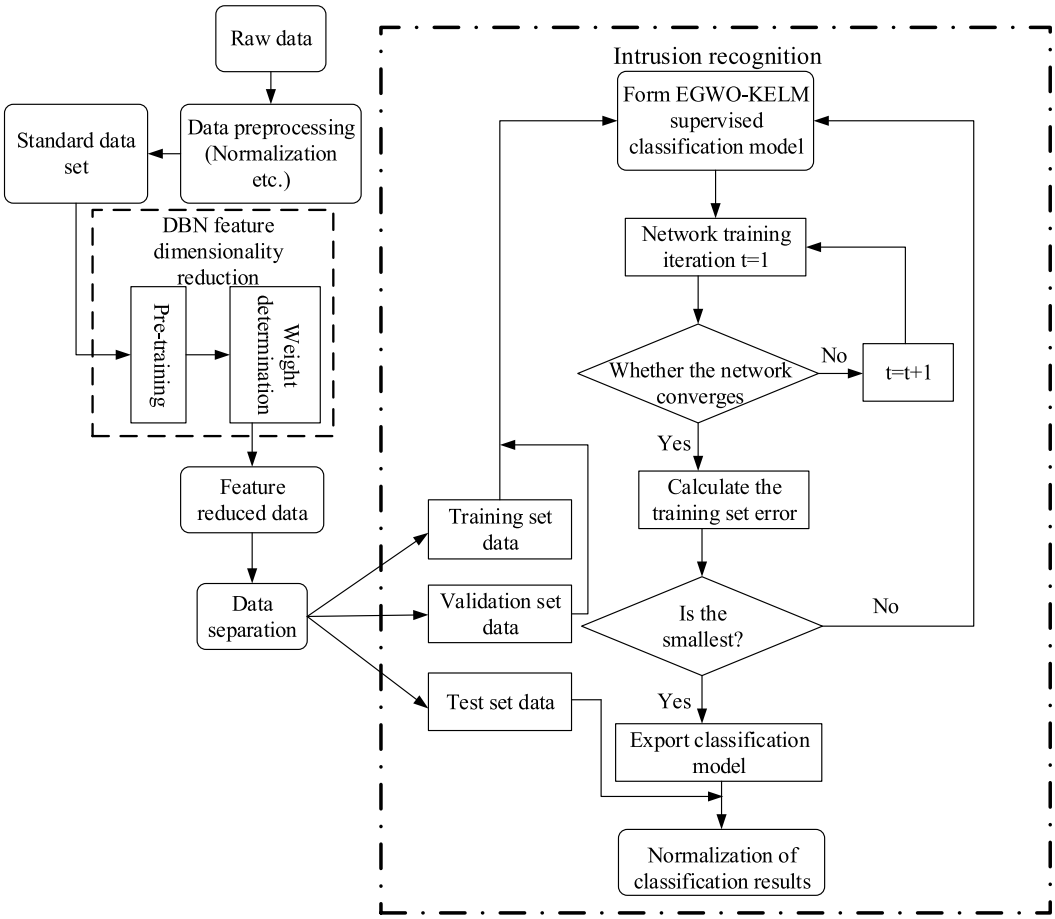


FIGURE 12. Flow chart of DBN-EGWO-KELM intrusion detection.

TABLE 3. Enhanced grey wolf optimizer parameters.

Parameters	Values	Parameters	Values
Population size	30	Inertia weight	1.2
Maximum number of iterations	500	Self-learning factor	2.4
Shrinkage factor	2-2*(t/T)	Group learning factor	1.6
Convergence accuracy	10e-8	Penalty factor	0.5

between the datasets. In the training stage, the Validation is used by the DBN for verification, and the error is calculated by the loss function after the validation. In the verification stage, the 6-step verification method and training accuracy limitation method described in the reference [47] are used to evaluate the training results, which is followed by the testing stage. The experimental results in Table 4 and Fig. 13 show that the number of iterations is 55 and the number of hidden layers is 80 when the model performance is the best. When the number of hidden layers increases, the training time increases and the amount of calculation will increase as well.

B. EVALUATION INDICATORS

In the binary classification experiment, the attacks in the dataset are merged as Abnormal and marked as 2, and Normal data is marked as 1. The intrusion detection accuracy rate (Acc), precision rate (P), true positive rate (TPR), false positive rate (FPR), F-score, Recall and other indicators are used for the classification experiment to make an evaluation. The indicator description is shown in Fig. 14, and the calculation method is referred to [48], [49].

C. MODELLING THE DATASET

Due to security and privacy issues, most datasets are not public. In addition, the public data has undergone painstaking anonymization, without considering the diversity of current network traffic. We consider the advantages and disadvantages of the existing dataset used in NIDS and discuss how our dataset is modeled.

1) KDDCup99: KDDCup99 is constructed by processing the tcpdump data of the 1998 DARPA Intrusion Detection Challenge dataset. The detailed statistical information of the dataset is shown in Table 5. The KDDCup1998 dataset was created by the Lincon Laboratory of the Massachusetts Institute of Technology using 1,000 UNIX machines and 100 users accessing these machines. The network traffic data was captured and stored in tcpdump format for 10 weeks. The first 7 weeks of data are used as the training dataset, and the rest of the data are used as the test dataset. The

TABLE 4. Results of parameter seeking experiment.

Hidden layers	Running time	Number of iterations	Cross entropy	Training set Accuracy rate (%)	Validation set accuracy rate (%)	Test set accuracy rate (%)
60	21s	206	0.0044	98.5	98.6	98.6
70	9s	69	0.0139	99.2	99.2	99.1
80	6s	55	0.0069	99.8	99.5	99.7
90	10s	73	0.0121	99.6	99.5	99.2
100	19s	139	0.0032	99.5	99.5	99.6
110	22s	78	0.0067	99.7	99.6	99.4

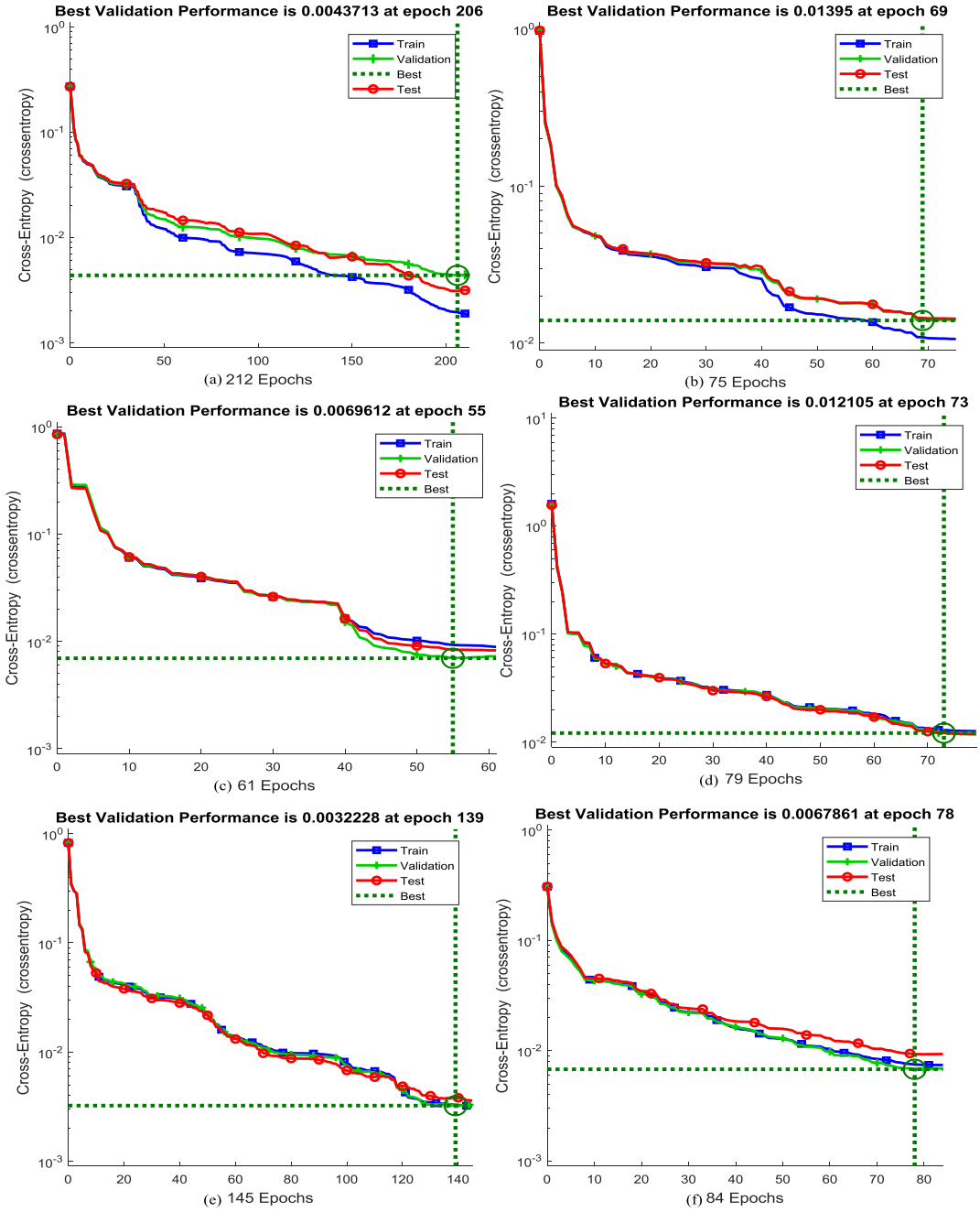


FIGURE 13. Network performance diagram.

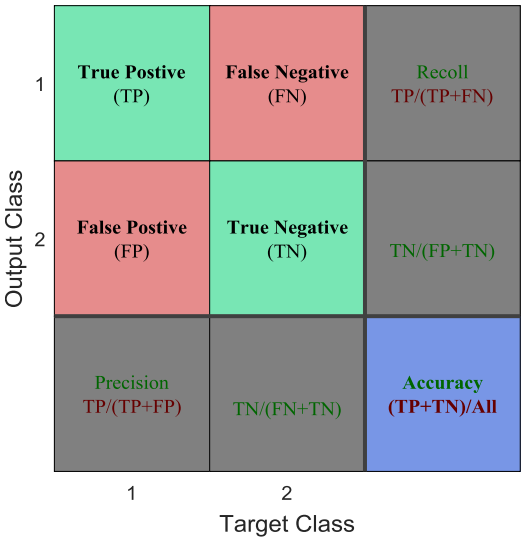


FIGURE 14. Structure evaluation index.

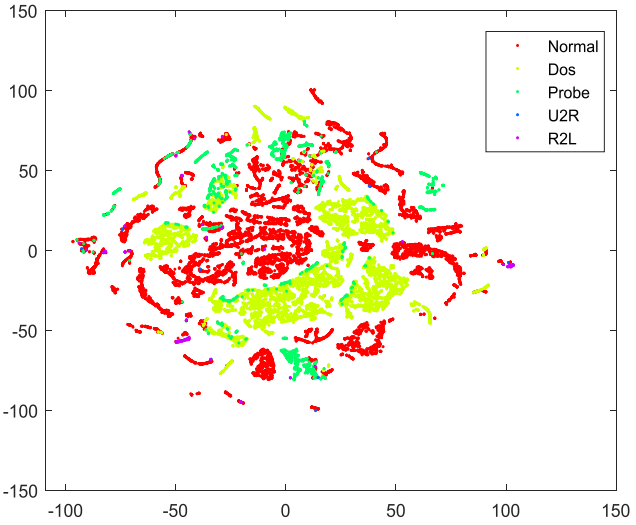


FIGURE 15. t-SNE visualization of KDDCup99.

KDDCup99 dataset that we used contains 8×10^4 pieces of network data information, 39 types of network attacks, and each piece of network data has 41 characteristic attributes and 1 class identifier. The data information includes 1 normal identification type Normal and 4 abnormal identification types Dos, Probe, U2R, R2L. Four anomalies contain a total of 22 attack types. These characteristics can be divided into the following different categories.

- Basic features: The packet capture (Pcap) files of tcpdump are used to extract the basic features from the packet headers, TCP segments, and UDP datagram instead of payload. This task was carried out using a remodeled network analysis framework, Bro IDS.
- Content features: Content features are extracted from the full payload of TCP/IP packets rooted on domain knowledge in tcpdump files. The feature analysis of payload has remained as research areas for the recent years. Wang introduced a deep learning method to analyze the entire payload data instead of following the feature extraction process [50]. Content features are mainly used to identify R2L and U2R attacks.
- Time-based traffic features: Time-based traffic features are extracted with a specific time window of 2 seconds. They are classified into "same host" and "same service" based on the connection characteristics in the past 2 seconds. To deal with slow detection attacks, the above characteristics will be recalculated based on a connection window of 100 connections to the same host. These are usually called connection-based or host-based traffic characteristics.

2) NSL-KDD is the essence of KDDCup99 invasion data. The filter is used to delete redundant connection records in KDDCup99, and delete connection records numbered 136,489 and 136,497 from the test data. NSL-KDD can protect machine learning algorithms from bias. Compared with the KDDCup99 dataset, this dataset is very suitable for misuse detection. It also reflects the characteristics of real-time network traffic profile. The NSL-KDD dataset we used has a total of 11,850 network data, and the data characteristics and attack types are consistent with the KDDCup99 dataset; The detailed statistics of NSL-KDD are shown in Table 5.

3) UNSW-NB15: The cyber security research team of Australian Centre for Cyber Security (ACCS) has introduced a new dataset called UNSW-NB15 to resolve the issues found in the KDDCup99 and NSL-KDD datasets. This dataset is generated in a mixed way, including normal attack behaviors and real-time network traffic using IXIA Perfect Storm tool, a repository of new attacks and common vulnerability exposures (CVE), including information security vulnerability libraries and exposures, which are made public. Two servers are used in the IXIA traffic generator tool. One server generates normal activities, while the other server generates malicious activities in the network. The tcpdump tool is used to capture network packet traces, the tool is also used to compile all 100 GBs of data into 1000 MB pcaps, which takes several hours. The Argus and bros-ids are used to extract features from the pcap file in Linux Ubuntu 14.0.4. In addition to the above methods, 12 algorithms developed by C# are used to perform in-depth analysis on each data packet. The UNSW-NB15 dataset we used has 175,300 connection data, and each data contains 49 features. In addition to normal data, it also contains 9 types of attacks including Fuzzers, Analysis, Backdoors, Dos, Exploits, Generic, Reconnaissance, Shellcode and Worms. Table 6 describes the types of simulated attacks and their detailed statistics.

4) CICIDS2017: This dataset contains benign attacks and describes real-time network traffic. The main interest is to collect real-time background traffic by creating this dataset, and use the B-profile system to collect benign background traffic. The benign traffic contains the features of 25 users

TABLE 5. Training, validation and testing connection records from KDDCup99 and NSL-KDD datasets.

Attack category	Description	Data instances					
		KDDCup99			NSL-KDD		
		Train	Validation	Test	Train	Validation	Test
Normal	Normal connection records	7125	2906	4175	923	382	568
Dos	Attacker aims at making network resources down	3112	1222	1907	1273	496	713
Probe	Obtaining detailed statistics of system and network configuration details	19311	7718	11486	2112	882	1323
R2L	Illegal access from remote computer	189	70	119	92	36	64
U2R	Obtaining the root or super-user access on a particular	10263	4048	6313	1525	574	887
Total		40000	16000	24000	5925	2370	3555

TABLE 6. Training, validation and testing connection records of partial dataset of UNSW-NB15.

Class	Description	Train	validation	Test
Normal	Normal connection records	26906	10920	16327
Fuzzers	Attacks related to spam html files penetrations and port scans	9537	3707	5510
Analysis	Attacks related to port scan, html file penetrations and spam	159	59	89
Backdoors	Backdoors is a mechanism used to access a computer by evading the background existing security	2	2	0
Dos	Intruder aims at making network resources down and consequently, resources are inaccessible to authorized users	1138	458	639
Exploits	The security hole of operating system or the application software is understand by an attacker with the aim to exploit vulnerability	24806	9988	14858
Generic	Attacks are related to block-cipher	19652	7777	11764
Reconnaissance	A target system is observe by an attacker to gather information for vulnerability	5309	2105	3317
Shellcode	A small part of program termed as payload used in exploitation of software	137	44	86
Worms	Worms replicate themselves and distributed to other system through the computer network	4	0	0
Total		87650	35060	52590

based on HTTP, HTTPS, FTP, SSH and email protocols. The network traffic is collected for 5 days, with the normal active traffic discarded on one day, and the attack injected on another day. The various attacks injected are Brute Force FTP, Brute Force SSH, Dos, Heartbleed, Web Attack, Infiltration, Botnet and DDoS. The CICIDS 2017 dataset contains 103485 pieces of network data information. The detailed information of the CICIDS 2017 dataset is shown in Table 7.

We have randomly selected all connection records in the NIDS dataset and passed them to t-SNE. The visual representations of KDDCup99, NSL-KDD, UNSW-NB15 and CICIDS 2017 are shown in fig 15-18. The connection record of CICIDS 2017 is more complicated than UNSW-NB15.

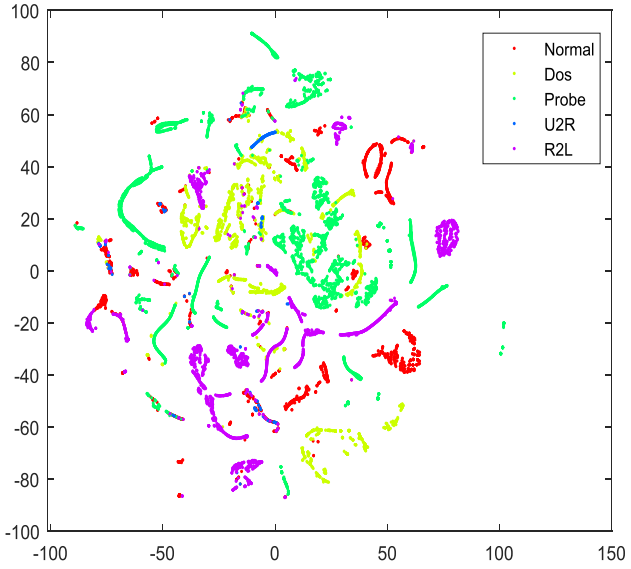


FIGURE 16. t-SNE visualization of NSL-KDD.

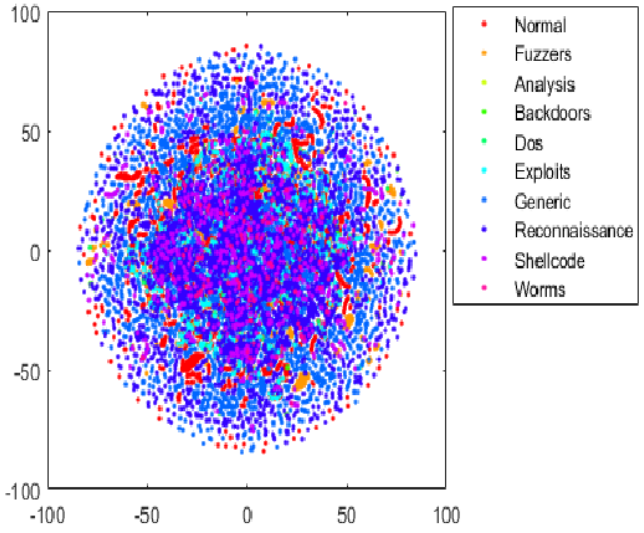


FIGURE 17. t-SNE visualization of UNSW-NB15.

In addition, the CICIDS 2017 dataset is recently released and contains new attacks. In addition, the CICIDS 2017 dataset has the characteristics of real-time network traffic, therefore, the proportion of normal data is larger.

D. BINARY CLASSIFICSTION EXPERIMENT

The experiment is conducted on the KDDCup99 dataset, NSL-KDD dataset, UNSW-NB15 dataset and CICIDS2017 dataset for intrusion testing. In the experiment process, the BP model, DBN model, DBN-KELM model, classic machine learning model, Multi-classifier LIBSVM, and CNN are compared with the DBN-EGWO-KELM model we proposed to compare and verify the superior performance of this algorithm. It can be seen from the visualization of t-SNE of the four datasets, the UNSW-NB15 and the CICIDS2017 dataset

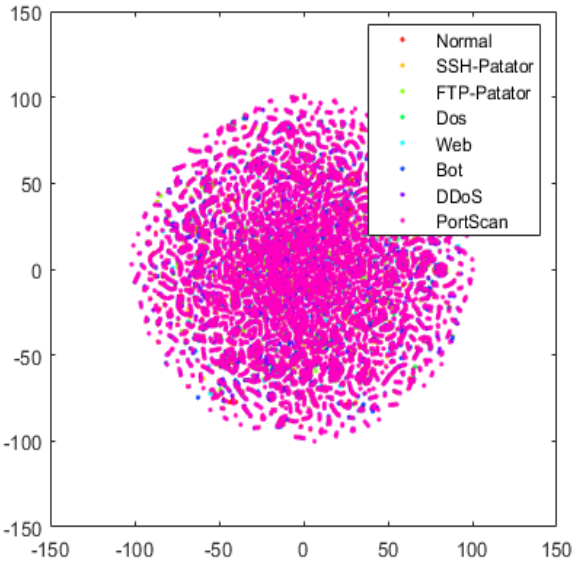


FIGURE 18. t-SNE visualization of CICIDS2017.

is relatively scattered and difficult to classify. Fig. 19-22 shows the confusion matrix of the binary classification experiment results of the DBN-EGWO-KELM algorithm model on the four datasets, using the format defined in Fig.14. Table 8 and Table 9 show the detailed results of binary classifications of each algorithm model. It is worth noting that there are many new types of attacks in CICIDS2017, and they are obtained from real-time traffic, and the proportion of normal data is large.

From the confusion matrix diagram of the binary classification results, we can see that for the KDDCup99, NSL-KDD, UNSW-NB15 and CICIDS2017 datasets, the training accuracy and test accuracy of the DBN-EGWO-KELM classification model are between 93% and 99%, the accuracy rate on the KDDCup99 and NSL-KDD datasets exceeds 98%. Tables 8 and 9 are detailed results of various machine learning classifiers, classic neural network algorithms and DBN-EGWO-KELM binary classification. It can be seen from Table 8 that in terms of training accuracy, the performance of the SVM classifier is better than the BP algorithm, RBF classifier and KELM. In addition, the performance of the SVM classifier maintains the same range on different datasets and maintains a high accuracy, however, the accuracy of the DBN-EGWO-KELM classification model compared with other methods is the highest and stable on each dataset, which is 5.58-29.51% higher. In terms of training accuracy, on the UNSW-NB15 dataset, the accuracy of the DBN-EGWO-KELM classification model is 82.54%, and the remaining algorithms are all below 80%. On the KDDCup99, NSL-KDD and CICIDS2017 datasets, the accuracy rates of the model proposed in this paper are as high as 93.50%, 95.30% and 96.63%, while the accuracy rates of the BP algorithm and the KELM algorithm are between 60-76%, however, the precision rates of RBF, SVM, DBN and DBN-KELM are all higher than 80%, and the best performance is the

TABLE 7. Training, validation and testing connection records of partial dataset of CICIDS2017.

Class	Description	Train	validation	Test
Normal	Normal connection records	29813	11981	17594
SSH-Patator	Secure shell-Representation of brute force attack	3860	1506	2372
FTP-Patator	File transfer protocol-Representation of brute force attack	4876	2027	3017
Dos	Intruder aims at making network resources down and consequently, resources are inaccessible to authorized users	630	245	336
Web	Attacks are related to web	1025	379	646
Bot	Hosts are controlled by bot owners to perform various tasks such as steal data, send spam and others	3598	1432	2166
DDoS	Distributed Denial of service('DDoS') is an attempt made to make service down using multiple sources. These are achieved using botnet	2949	1119	1810
PortScan	Port scan is used to find the specific port which is open for a particular service. Using this attacker can get information related to sender and receiver's listening information	4994	2007	3103
Total		51745	20696	31044

FIGURE 19. Experimental results of binary classification on KDDCup99.

DBN-KELM model. The precision rates on the three datasets are 91.55%, 92.47%, and 90.22% respectively. In terms of training recall rate, the BP algorithm performed the worst on the four datasets, with recall rates of 75.21%, 79.65%, 69.24%, and 70.24% respectively. The DBN-EGWO-KELM classification model proposed in this paper performs best on the four datasets, with recall rates of 98.90%, 98.70%, 96.60%, and 98.32% respectively. The other algorithms in recall rates can reach more than 80% on both KDDCup99 and NSL-KDD datasets. On the UNSW-NB15 dataset, the recall rates of the RBF classifier and KELM are 77.94% and 79.54%, respectively. The recall rates of the other algorithms are all higher than 80%. Compared with the DBN-KELM model, the DBN-EGWO-KELM classification model proposed in this paper improves 7.28%. When the precision rate and the recall rate conflict, it will be much more difficult to compare the performance of the model, and the F-score can take into account both the precision rate and the recall

FIGURE 20. Experimental results of binary classification on NSL-KDD.

FIGURE 21. Experimental results of binary classification on UNSW-NB15.

rate, which can be regarded as a harmonic average to better evaluate the model. In Table 8, the F-score of the DBN-EGWO-KELM classification model on the four datasets are 96.12%, 96.97%, 89.02%, and 97.48%, respectively, which

FIGURE 22. Experimental results of binary classification on CICIDS2017.

are improved by 4.7%, 4.47%, 5.03%, 8.78% compared to the sub-optimal DBN-KELM model. It can indicate that the DBN-EGWO-KELM classification model not only shows high classification performance on each dataset, but also has high stability.

It can be seen from Table 9 that on the test set, the classification accuracy, precision and recall rate of the DBN-EGWO-KELM classification model are all higher than other algorithms. In the KDDCup99 and NSL-KDD datasets, it can reach more than 94%. Due to the large amount of data in the UNSW-NB15 dataset, it can also be seen through t-SNE visualization that the data in the dataset is relatively scattered, but the DBN-EGWO-KELM classification model still maintains its superior performance, with the accuracy rate of 93.42%, the precision rate is 82.30, the recall rate is 96.40%, and the F-score is 88.79%. CICIDS2017 is a relatively new dataset. Although the amount of data is large, it can be seen from the t-SNE visualization that the dataset is relatively scattered, but the DBN-EGWO-KELM classification model can still output very good results, and the classification accuracy rate is as high as 97.07%. The time spent is only 202s, which can reflect that the data processing and classification performance of our proposed model is superior to the classic BP algorithm and machine learning classifier. The experimental results show that the SVM classifier performs well, on the KDDCup99 and NSL-KDD datasets, each evaluation index can reach more than 85%, and it has a high evaluation on the UNSW-NB15 dataset. It shows that the SVM method is suitable for binary classification problems, but the performance of deep neural networks is significantly better than the classic machine learning algorithms, and has great advantages in binary classification.

E. MULTI-CLASSIFICATION EXPERIMENTS

Fig 23-26 shows the confusion matrix of the results of the DBN-EGWO-KELM algorithm model on the four datasets. Tables 10 and 11 show the detailed results of multi-classifications of each algorithm model.

It can be seen from Table 10 that the accuracy of the DBN-EGWO-KELM classification model on the four training datasets is the highest 96.5%, 94.0%, 79.5%,97.2%, respectively. For other algorithms such as BP, RBF, SVM, LIBSVM, KELM, CNN, DBN and DBN-KELM, the accuracy rates on the KDDCup99 dataset are 73.3%, 78.6%, 80.2%, 87.4%, 79.9%, 89.4%, 86.5%, 88.7%, and the accuracy rates on the NSL-KDD dataset are 73.0%, 77.1%, 80.2%, 81.6%, 76.8%, 88.4%, 87.1%, 89.2%, the accuracy rates on the UNSW-NB15 dataset are 50.9%, 45.6%, 52.9%, 55.4%, 65.9%, 70.2%, 68.6%, 75.6%, and the accuracy rates on the CICIDS2017 dataset are 80.9%, 82.9%, 85.4%, 88.3%, 85.8%, 82.1%, 84.9%, 90.8%. It is obvious that the accuracy of each algorithm on the UNSW-NB15 dataset is lower than

TABLE 8. Experimental results of binary classification of each algorithm on training set.

Algorithm	Acc(%)	P(%)	Recall(%)	F-score (%)	Time(s)
KDDCup99					
BP	77.16	69.51	75.21	72.23	3214
RBF	82.59	80.92	81.26	81.09	2418
SVM	85.22	90.64	86.24	88.39	646
LIBSVM	84.04	89.54	85.93	87.70	518
KELM	81.46	69.62	80.56	74.69	588
CNN	83.58	79.52	70.68	74.42	1770
DBN	86.77	80.23	86.24	83.13	1264
DBN-KELM	92.56	91.55	91.29	91.42	449
DBN-EGWO-KELM	98.50	93.50	98.90	96.12	206
NSL-KDD					
BP	79.36	72.24	79.65	75.76	618
RBF	82.98	82.19	81.36	81.77	426
SVM	87.21	92.13	87.36	89.68	98
LIBSVM	88.39	87.26	85.77	86.51	106
KELM	83.82	75.41	83.24	79.13	168
CNN	84.57	88.24	83.56	85.84	420
DBN	85.23	80.36	84.69	82.47	365
DBN-KELM	93.32	92.47	92.54	92.50	227
DBN-EGWO-KELM	98.90	95.30	98.70	96.97	63
UNSW-NB15					
BP	70.48	60.54	69.24	64.59	5628
RBF	78.25	65.85	77.94	71.38	4652
SVM	84.32	75.22	85.62	80.08	1258
LIBSVM	84.56	78.69	80.47	79.57	1005
KELM	80.98	69.36	79.54	74.10	1154
CNN	79.45	77.26	80.24	78.72	3521
DBN	82.36	72.63	82.26	77.15	2894
DBN-KELM	88.37	79.26	89.32	83.99	995
DBN-EGWO-KELM	93.54	82.54	96.60	89.02	393
CICIDS2017					
BP	67.56	60.23	70.24	64.85	3015
RBF	74.24	82.47	79.17	80.79	2564
SVM	88.29	83.14	86.45	84.76	654
LIBSVM	90.14	80.26	82.44	81.34	542
KELM	88.11	75.89	83.46	79.50	655
CNN	84.12	70.39	84.87	76.95	1574
DBN	86.47	80.36	88.29	84.14	1486
DBN-KELM	90.22	85.22	92.47	88.70	489
DBN-EGWO-KELM	97.07	96.63	98.32	97.48	202

the other two datasets. From the t-SNE visualization of the dataset in Fig 15-18, it can be seen that the data of the UNSW-NB15 dataset is scattered, there are many types of attacks, and the amount of data is huge, which affects the classification effect of each algorithm. Many datasets are unbalanced in the sense that the sample size of each category in a dataset is not equal. Taking the binary classification problem as an example. Assuming that the number of positive samples is far greater than the number of negative samples, then the proportion of the major versus minor samples is close to 100:1, this dataset is called an unbalanced dataset. The learning of unbalanced dataset is to learn useful information from non-uniform datasets. For these unbalanced datasets, the intrusion detection performance of each algorithm will be further analyzed from the view of the precision, false positive rate, and true positive rate.

It can be seen from Table 10 that in the KDDCup99 dataset, for the Normal type of data, the detection performance of BP, RBF and KELM is the worst, the true positive rate is only 3.9%, 3.6% and 2.3%, compared to DBN-EGWO-KELM classification model the rate differs by 93.3%, 93.6% and 94.9%., It is obvious that the DBN-EGWO-KELM

TABLE 9. Experimental results of binary classification of each algorithm on test set.

Algorithm	Acc(%)	P(%)	Recall(%)	F-score (%)	Time（s）
KDDCup99					
BP	79.57	70.25	79.23	74.47	1751
RBF	83.66	81.36	83.84	82.58	1325
SVM	86.58	89.41	85.99	87.67	313
LIBSVM	85.36	88.24	86.32	87.27	259
KELM	83.37	72.56	82.45	77.19	254
CNN	85.79	87.24	83.21	85.18	905
DBN	88.29	79.89	87.25	83.41	618
DBN-KELM	93.56	90.21	93.54	91.84	247
DBN-EGWO-KELM	98.60	94.00	98.73	96.31	134
NSL-KDD					
BP	80.10	72.53	79.78	75.98	314
RBF	84.28	85.22	82.66	83.92	203
SVM	89.42	92.32	88.46	90.35	54
LIBSVM	85.63	89.24	86.32	87.76	59
KELM	84.28	78.15	82.53	80.28	95
CNN	86.24	84.23	86.24	85.18	222
DBN	87.86	82.67	86.70	84.64	189
DBN-KELM	93.12	93.55	92.20	89.87	112
DBN-EGWO-KELM	98.60	93.64	98.40	96.06	41
UNSW-NB15					
BP	70.89	59.90	68.78	64.03	2641
RBF	78.55	66.27	76.24	70.91	2248
SVM	85.12	75.25	85.94	80.24	655
LIBSVM	82.34	76.34	70.22	73.15	521
KELM	80.98	68.98	79.33	73.79	589
CNN	80.11	81.29	78.22	79.73	1742
DBN	81.69	73.13	82.10	77.36	1452
DBN-KELM	88.21	80.61	88.34	84.30	552
DBN-EGWO-KELM	93.42	82.30	96.40	88.79	214
CICIDS2017					
BP	72.56	63.24	70.11	66.50	1541
RBF	80.12	66.98	78.98	71.99	1258
SVM	87.36	76.21	89.47	82.31	309
LIBSVM	83.55	74.28	73.29	73.78	267
KELM	82.17	70.32	78.22	74.06	345
CNN	81.22	83.49	83.47	83.48	772
DBN	82.39	75.68	84.55	79.87	785
DBN-KELM	90.95	83.69	89.27	86.39	251
DBN-EGWO-KELM	97.15	96.80	98.19	97.49	120

classification model performs best under this label. It can be seen from Fig 24 that the DBN-EGWO-KELM classification model misjudged 1152 attacks as normal data, resulting in slightly worse accuracy and false positive rates than DBN-KELM, which were 85.7% and 3.5%. For Dos attacks, the DBN-EGWO-KELM model has the best detection performance, with a true positive rate of 98.0%, and KELM has the worst detection performance with a true rate of 54.2%. The performance evaluation indicators of other methods are good. Dos account for the largest proportion, and need to be processed separately when judging by clustering. Since this article is based on the difference between normal data and abnormal data, Dos attacks with a large amount of data can still maintain good detection results. For Probe attacks, the performance of BP, KELM, DBN, and DBN-KELM are similar and far better than RBF, but overall, the performance of the DBN-EGWO-KELM model is the best, with the true positive rate of 99.9% and accuracy of 99.4%. For R2L attack types, only DBN-KELM and DBN-EGWO-KELM perform well, and the other methods have similarly poor performance. Among them, the SVM classifier has the worst detection performance and basically cannot identify R2L attacks, because

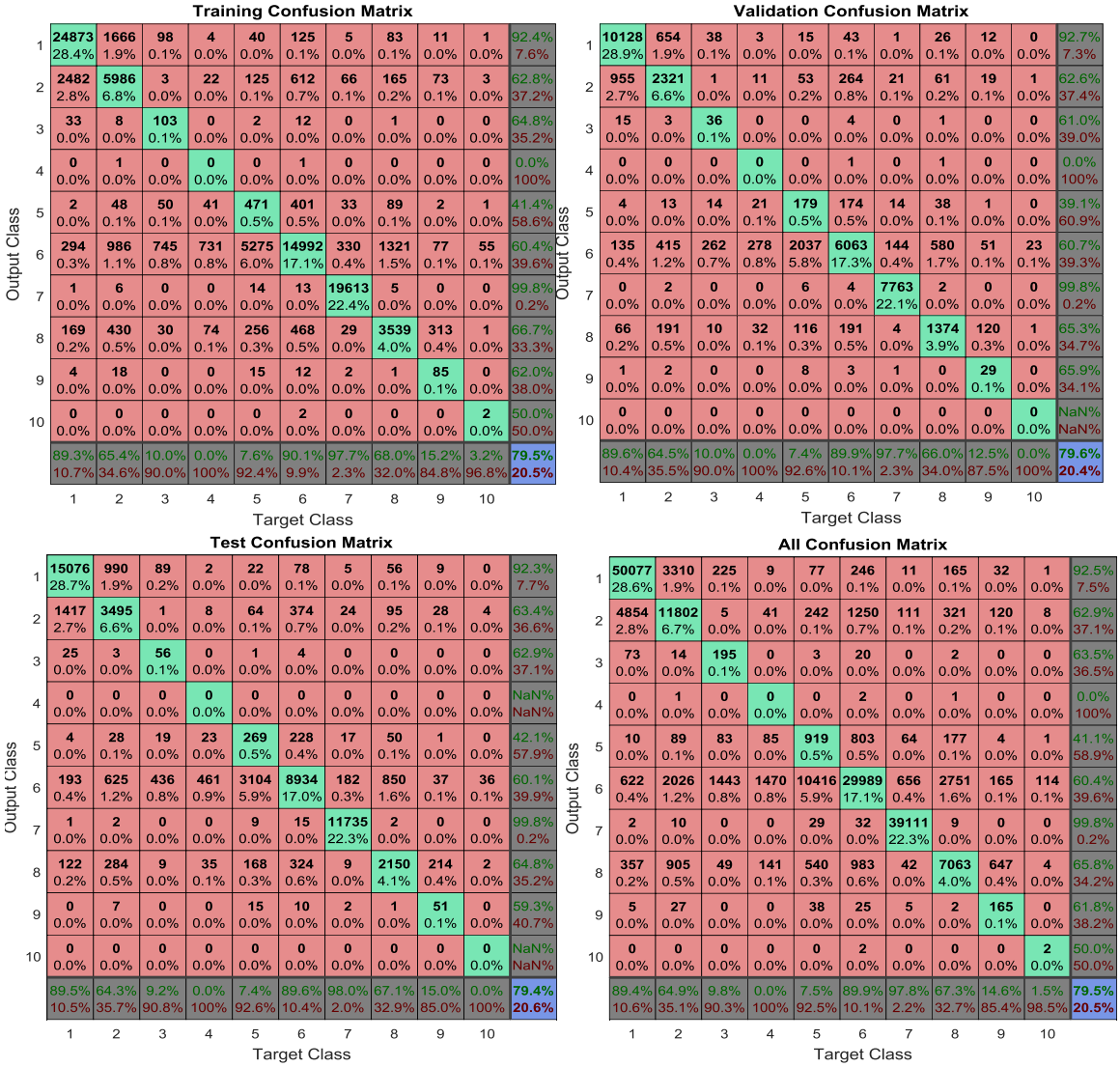


FIGURE 23. Multi-classification experimental results on UNSW-NB15.

the sample size of R2L attacks in the training set is very small, and the R2L attack is carried out by disguising as a legitimate user, which is similar to the normal data characteristics, making detection difficult. However, the DBN-EGWO-KELM model learns the characteristics of R2L well, and correctly classify it, which is also reflected in the test set in Table 11. For U2R attacks, RBF, KELM, DBN, and DBN-KELM have similarly good performance. The true positive rates are all above 94%. The true positive rates of BP, SVM and DBN-EGWO-KELM are all above 88%. Although the detection capability of DBN-EGWO-KELM model is comparable with other algorithms in this test, its overall detection performance is outstanding and meets expectations.

From the results of the multi-classification experiment on the NSL-KDD dataset in Table 10, it can be seen that the detection performance is slightly lower than that of the KDD-Cup99 dataset. This is because the amount of NSL-KDD sample data is small and the amount of data available for training is relatively small. The DBN-EGWO-KELM model has the highest true positive rates on Normal, Dos, Probe, and R2L than other methods, which are 92.2%, 96.5%, 98.1%, and 90.2%, respectively. The false positive rates are as low as 4.9% and 0.5%, 1.1%, 0.2%. In U2R attacks, BP, RBF and DBN-EGWO-KELM have similar performance, and the true positive rate is worse than SVM, KELM, DBN and DBN-KELM, however, the DBN-EGWO-KELM model has the lowest false positive rate of 1.1%, which is 39.1% lower than the RBF with the largest false positive rate.

According to the detailed results of multi-classification on the UNSW-NB15 dataset in Table 10, it can be seen that for Normal data, the detection performance of BP and RBF is at an average level, and the true positive rates are only 52.7% and 56.6%, compared with the DBN-EGWO-KELM classification model, the true positive rate is 39.7% and 35.8% lower. For the other algorithms, the true positive rate of KELM is 81.6%; The true positive rate of LIBSVM is 82.6%,

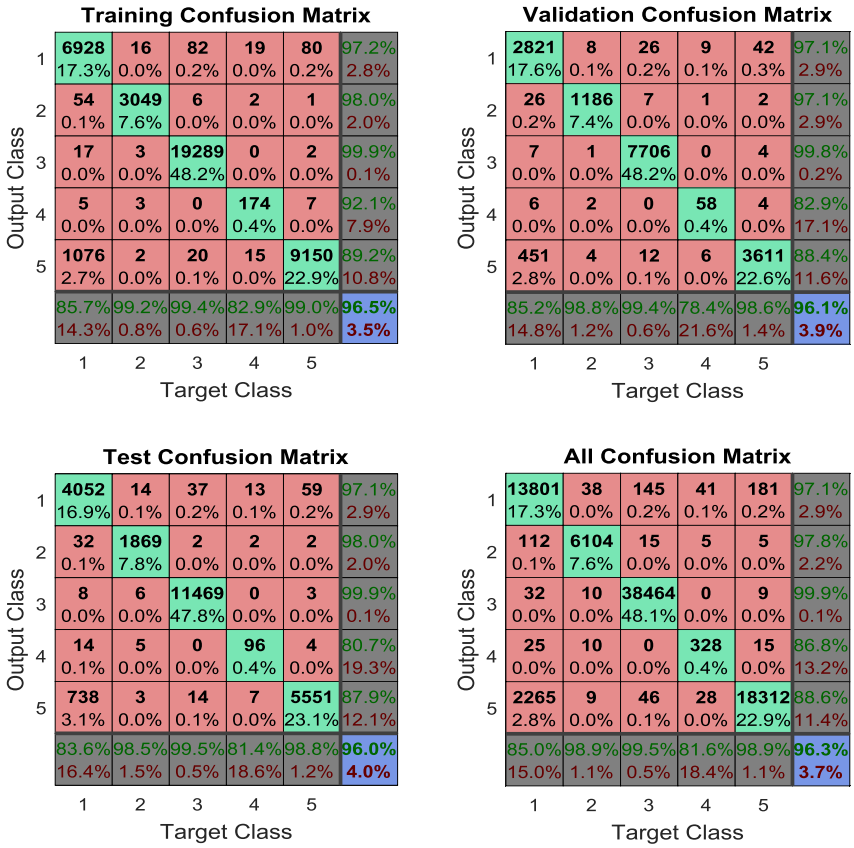


FIGURE 24. Multi-classification experimental results on KDDCup99.

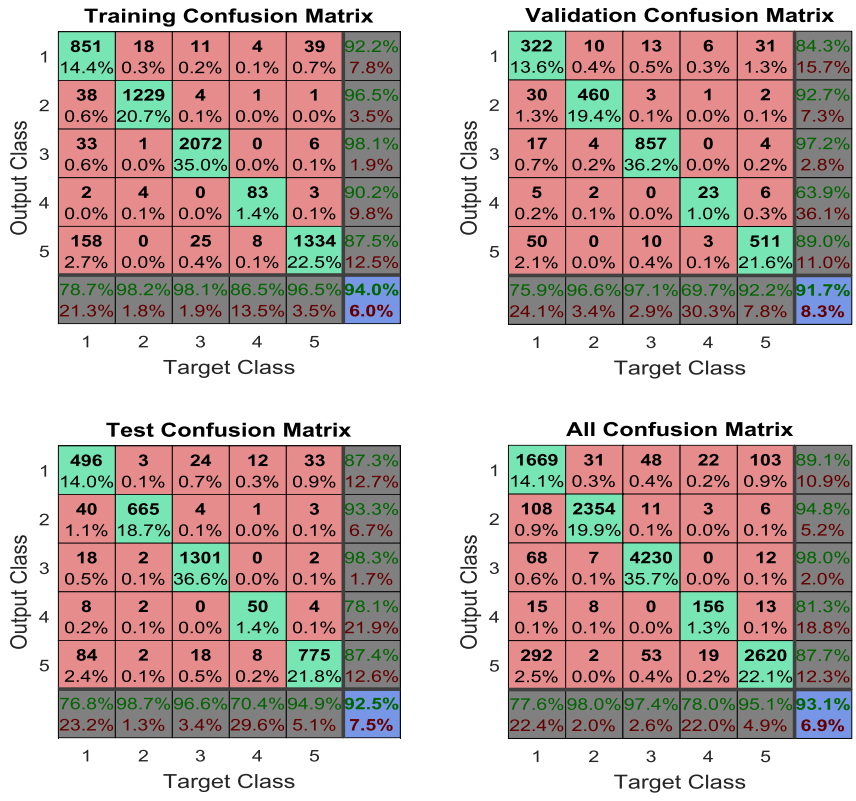


FIGURE 25. Multi-classification experimental results on NSL-KDD.

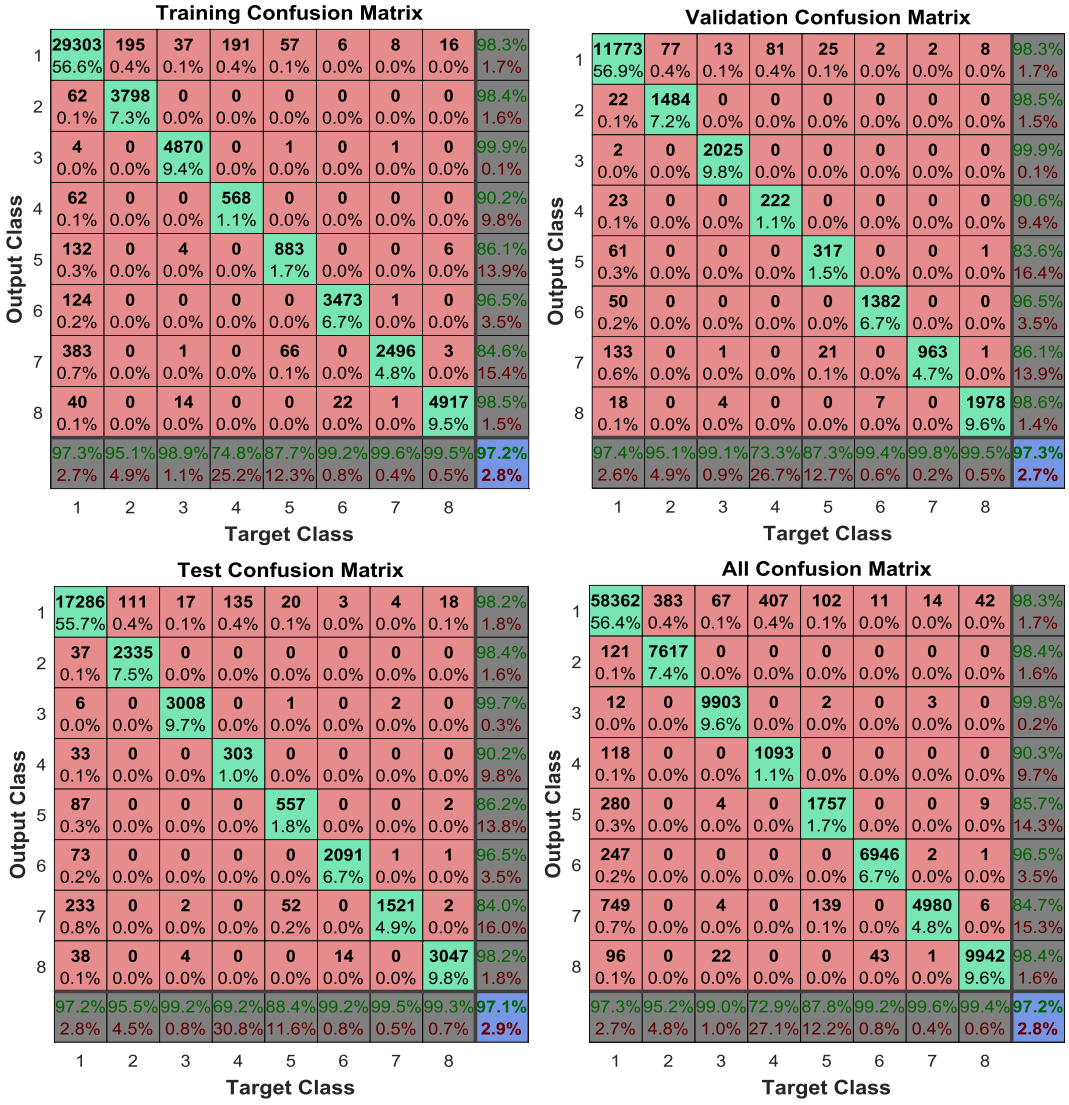


FIGURE 26. Multi-classification experimental results on CICIDS2017.

the rest are around 90%. The DBN-EGWO-KELM model has the highest true positive rate, with the false positive rate as low as 6.2%, and the accuracy rate is the highest at 89.3%. For Fuzzers and Analysis attacks, the DBN-EGWO-KELM model also has the best detection performance, not only the highest true positive rate, but also the lowest false positive rate. For Backdoors attacks, since the attack sample size only accounts for 1% of the total sample size, the performance of the classification performance of all algorithms in this attach is very poor. The true positive rate of most of the algorithms is 0, while the highest is SVM, which is only 27.6%. For Dos attacks, the performance of each algorithm in the evaluation index is at an average level, while the performance of the DBN-EGWO-KELM model is the best. For Exploits attacks, the performance of BP, RBF and KELM are similar, and worse than SVM, DBN, DBN-KELM and DBN-EGWO-KELM. Among them, DBN-EGWO-KELM has the best performance, with the true positive rate of 60.4%, and the false positive rate only 2.9%, which is 30.9% lower than RBF. For Generic attacks, the true positive rate of the DBN-EGWO-KELM model is as high as 99.8%, whose performance is much better than other algorithms, nearly 20% higher than the sub-optimal DBM-KELM model. Its false positive rate is the lowest, compared to BP, RBF, SVM, KELM, DBN, and DBN-KELM are 1%, 7.5%, 2%, 3%, 16.7%, 14.6% lower, respectively, and the accuracy of the DBN-EGWO-KELM model is the highest at 97.7%. For the Reconnaissance attack, BP and RBF failed to detect the attack. SVM and DBN have similar performance, but are lower than KELM and DBN-KELM, which have similar performance. The best performance is still the DBN-EGWO-KELM model. For Shellcode attacks, since the proportion of data in the attack is less than 1%, the true positive rates of BP, RBF, SVM, KELM, DBN, and DBN-KELM are all lower than 60%. Among them, the best is DBN-KELM, the true positive rate is 58.9%, and the false positive rate is 1.3%. Only the true positive rate of the

TABLE 10. Experimental results of multi classification of each algorithm on training set.

Algorithm	Normal			Dos			Probe			R2L			U2R			
KDDCup99																
Unit(%)	TPR	FPR	P	TPR	FPR	P	TPR	FPR	P	TPR	FPR	P	TPR	FPR	P	Acc
BP	3.9	0.3	75.2	81.5	5.0	93.1	97.5	6.7	82.3	31.3	0.2	87.5	90.2	28.5	52.3	73.3
RBF	3.6	0.4	74.3	90.1	0.5	99.0	59.6	0.07	99.3	56.7	0.5	68.0	94.1	16.6	67.2	78.6
SVM	63.7	1.5	90.3	90.2	12.3	86.2	88.8	9.4	75.4	0	0	NaN	88.7	4.5	87.9	80.2
LIBSVM	70.2	4.9	92.3	92.3	7.4	88.2	87.2	10.3	70.9	10.9	0.8	71.8	89.7	6.3	90.7	87.4
KELM	2.3	52.4	89.5	54.2	1.4	96.7	91.5	7.2	80.2	29.3	0.14	81.2	96.2	20.5	65.2	79.9
CNN	71.9	10.4	70.9	91.8	4.5	95.5	97.2	2.2	92.6	50.7	0.5	93.2	97.7	5.8	84.4	89.4
DBN	62.4	14.3	60.3	89.6	5.7	93.5	94.4	2.8	89.7	54.2	0.11	90.2	95.9	9.7	77.7	86.5
DBN-KELM	64.2	2.1	88.9	89.9	2.3	96.3	93.8	3.3	89.6	82.8	0.4	92.3	94.3	7.2	80.2	88.7
DBN-EGWO-KELM	97.2	3.5	85.7	98.0	0.06	99.2	99.9	0.55	99.4	92.1	0.09	82.9	89.2	0.3	99.0	96.5
NSL-KDD																
Unit(%)	TPR	FPR	P	TPR	FPR	P	TPR	FPR	P	TPR	FPR	P	TPR	FPR	P	Acc
BP	3.3	0.6	75.6	80.2	7.5	92.4	96.6	12.3	82.6	31.2	1.3	88.2	87.2	30.9	53.3	73.0
RBF	4.2	1.1	72.1	88.7	1.9	97.7	60.3	3.1	98.4	55.5	2.2	59.6	84.6	40.2	67.9	77.1
SVM	50.3	2.6	88.2	86.9	19.4	86.6	87.2	13.9	74.3	0	0	NaN	90.3	6.9	85.6	80.2
LIBSVM	58.3	3.3	90.8	88.7	9.3	89.7	90.8	15.4	80.9	55.2	2.2	69.5	88.1	17.5	78.5	81.6
KELM	1.2	54.3	85.6	56.6	6.2	96.8	88.2	8.9	79.5	28.3	3.0	70.6	95.6	30.2	64.3	76.8
CNN	70.2	10.8	62.7	91.5	3.3	95.4	89.4	1.5	89.4	63.4	2.6	85.9	95.7	14.2	79.4	88.4
DBN	60.4	16.7	55.3	90.2	6.7	92.1	89.9	3.2	85.3	49.3	1.3	82.3	94.2	13.5	75.4	87.1
DBN-KELM	65.3	3.5	80.9	85.6	3.8	97.2	93.6	4.2	89.1	81.5	0.9	86.6	95.6	7.3	78.9	89.2
DBN-EGWO-KELM	92.2	4.9	78.7	96.5	0.5	98.2	98.1	1.1	98.1	90.2	0.2	86.5	87.5	1.1	96.5	94.0
	Normal			Fuzzers			Analysis			Backdoors			Dos			
UNSW-NB15																
Unit(%)	TPR	FPR	P	TPR	FPR	P	TPR	FPR	P	TPR	FPR	P	TPR	FPR	P	
BP	52.7	2.5	72.9	27.6	6.8	55.4	44.7	56.6	33.9	0.0	0.0	20.9	10.6	24.0	19.6	
RBF	56.6	49.3	83.7	14.6	11.2	60.3	3.5	25.4	25.9	0.8	0.8	25.3	23.5	4.8	22.6	
SVM	89.2	28.9	76.9	55.7	9.5	59.2	55.5	36.8	44.8	27.6	3.2	18.9	30.9	2.5	34.6	
LIBSVM	82.6	30.5	80.7	33.9	25.3	63.9	69.4	23.7	55.7	22.1	2.9	20.4	44.2	5.4	30.9	
KELM	81.6	14.9	77.5	69.6	17.6	60.9	1.8	44.8	63.2	1.7	2.3	0.9	35.6	0.0	12.3	
CNN	88.6	20.4	81.2	59.6	15.7	65.2	24.9	68.4	30.1	0.0	1.9	10.3	33.4	0.8	29.1	
DBN	87.9	23.5	78.9	54.3	13.3	64.4	7.6	69.2	25.8	0.0	0.0	0.5	40.2	0.0	25.8	
DBN-KELM	91.5	26.4	89.0	55.6	22.3	65.2	29.6	41.5	36.8	0.0	0.0	0.0	39.5	0.0	30.9	
DBN-EGWO-KELM	92.4	6.2	89.3	62.8	4.7	65.4	64.8	1.3	10.0	0.0	1.2	0.0	41.4	7.6	7.6	
	Exploits			Generic			Reconnaissance			Shellcode			Worms			
Unit(%)	TPR	FPR	P	TPR	FPR	P	TPR	FPR	P	TPR	FPR	P	TPR	FPR	P	Acc
BP	45.6	0.0	78.9	50.9	1.9	80.7	0.0	0.0	NaN	44.5	5.3	3.3	12.9	5.9	0.03	50.9
RBF	46.8	33.8	75.6	61.6	8.4	78.8	0.0	0.0	NaN	23.6	8.3	4.2	25.6	30.6	0.2	45.6
SVM	55.6	27.1	82.7	70.9	2.9	81.1	33.8	5.3	55.6	50.3	3.9	6.9	33.5	21.3	0.9	52.9
LIBSVM	60.3	15.4	83.2	80.4	5.7	86.9	48.6	9.4	67.2	55.4	5.9	3.7	40.7	20.1	2.9	55.4
KELM	42.2	25.0	80.3	57.7	4.9	88.5	57.9	3.7	62.3	49.6	3.7	5.5	40.6	15.9	1.3	65.9
CNN	62.1	22.7	89.4	79.4	7.7	89.7	54.1	5.6	52.7	59.2	2.9	6.4	60.7	12.3	3.7	70.2
DBN	56.9	3.5	85.6	72.6	17.6	90.2	25.0	8.8	66.9	57.3	1.3	7.9	45.9	3.9	2.2	68.6
DBN-KELM	59.6	4.5	86.9	80.2	15.5	92.3	60.9	3.9	67.6	58.9	0.9	11.3	48.3	1.2	2.5	75.6
DBN-EGWO-KELM	60.4	2.9	90.1	99.8	0.9	97.7	66.7	2.4	68.0	62.0	0.68	15.2	50.0	0.08	3.2	79.5

TABLE 10. *(Continued.)* **Experimental results of multi classification of each algorithm on training set.**

	Normal			SSH-Patator			FTP-Patator			Dos			Web			
CICIDS2017																
Unit(%)	TPR	FPR	P	TPR	FPR	P	TPR	FPR	P	TPR	FPR	P	TPR	FPR	P	
BP	80.3	0.0	75.4	49.7	3.4	84.2	10.2	5.6	30.4	67.5	3.6	70.4	67.8	21.6	70.1	
RBF	81.6	26.3	70.4	50.9	12.3	81.4	25.9	0.0	56.4	74.2	31.6	77.4	44.8	40.8	72.5	
SVM	83.4	2.8	76.5	45.0	0.0	75.6	79.4	0.0	67.5	85.2	12.5	79.3	74.2	55.2	75.1	
LIBSVM	82.4	0.25	80.4	60.7	3.6	68.2	80.7	0.0	70.5	86.7	5.9	75.1	76.9	61.4	74.0	
KELM	70.8	55.3	76.5	0.0	0.0	NaN	72.4	0.04	68.1	74.6	0.0	70.5	71.5	65.2	79.6	
CNN	72.4	64.8	69.5	0.0	0.0	NaN	64.2	0.5	66.3	70.9	23.6	82.4	64.8	80.4	80.1	
DBN	79.4	45.2	75.9	56.4	2.1	67.2	73.9	1.6	75.2	78.8	17.9	80.6	75.2	89.4	82.4	
DBN-KELM	88.7	30.8	77.8	61.2	0.0	60.4	78.6	0.06	79.4	83.6	10.6	86.4	80.1	74.2	79.9	
DBN-EGWO-KELM	90.4	10.9	81.2	61.1	1.5	70.9	84.7	0.04	86.4	90.7	3.5	89.1	89.5	90.5	90.4	
	Bot			DDoS			PortScan									
Unit(%)	TPR	FPR	P	TPR	FPR	P	TPR	FPR	P							Acc
BP	66.7	0.0	54.4	49.5	3.6	77.6	69.2	5.6	40.4							80.9
RBF	34.5	24.3	69.2	45.9	22.1	80.4	70.6	0.3	52.4							82.9
SVM	75.8	2.7	72.1	36.0	0.3	73.6	79.4	2.0	66.6							85.4
LIBSVM	78.4	0.85	45.4	60.3	2.4	67.2	82.7	0.35	68.1							88.3
KELM	76.6	51.3	62.3	10.0	0.35	20.3	72.5	0.04	65.7							85.8
CNN	70.5	46.8	49.5	55.2	0.02	10.2	54.2	0.6	66.4							82.1
DBN	72.8	35.2	64.7	56.4	2.8	69.2	74.9	1.6	73.2							84.9
DBN-KELM	76.9	40.8	73.8	61.2	3.2	62.4	78.9	0.16	76.4							90.8
DBN-EGWO-KELM	80.7	10.2	84.3	61.1	0.5	73.9	86.7	0.3	87.9							97.2

DBN-EGWO-KELM model is higher than 60%, which is 62.0%, the false positive rate is the lowest, 0.68%. For Worms attacks, the true positive rate, false positive rate, and accuracy rate are all very low for all algorithms. Among them, the accuracy rate is the most obvious, all within 10%. Because the sample size of Worms attacks in the dataset is very small, only 130, however, DBN-EGWO-KELM model can still maintain the highest true positive rate, the lowest false positive rate, and the highest accuracy rate, indicating that the model we proposed can better learn the characteristics of the data and perform correct classification in a small amount of data.

It can be seen from Table 10 that on the CICIDS2017 dataset, the accuracy of all algorithms can reach more than 80%, which is very high. Although this dataset has a large amount of data and many new attacks, from the t-SNE visualization, there are more normal data and relatively few abnormal data, which leads to a reduction in classification difficulty. In this case, the new DBN-EGWO-KELM classification model still maintains the highest accuracy rate, regardless of the composition of the abnormal or normal data.

It can be seen from Table 11 that on the test set, the classification accuracy, true positive rate, and false positive rate of the DBN-EGWO-KELM classification model are higher than other algorithms. On the KDDCup99 and NSL-KDD datasets, they can reach 92 % or more, due to the large amount of data. In the UNSW-NB15 dataset, it can be seen through t-SNE visualization that the data is relatively scattered and there are many different types of attacks. Although the overall accuracy rate has declined, the DBN-EGWO-KELM classification model still maintains its superior performance, with an accuracy rate of 79.4%, which is 5% higher than the sub-optimal DBN-KELM model. In various attacks, the true positive rate of DBN-EGWO-KELM can be maintained at a high level, while the false positive rate can be maintained at a low level, and hence the accuracy rate is generally high. Compared with the training set, the evaluation indicators of the DBN-EGWO-KELM model are more stable on the four datasets and maintained at a high level.

The results of multi-classification experiments show that the performance of the DBN-EGWO-KELM model is the optimal in terms of accuracy, which is obviously better than the classical neural networks and the classical machine learning algorithms. The advantage of the proposed algorithm is also reflected in the rest of measures.

VII. CONCLUSION AND FUTURE WORK

We propose an intrusion detection method based on an improved deep belief network. A novel kernel extreme learning machine classification model is designed using enhanced grey wolf optimizer optimization, which extracts

TABLE 11. Experimental results of multi classification of each algorithm on test set.

Algorithm	Normal			Dos			Probe			R2L			U2R			
KDDCup99																
Unit(%)	TPR	FPR	P	TPR	FPR	P	TPR	FPR	P	TPR	FPR	P	TPR	FPR	P	Acc
BP	3.9	0.3	75.2	81.5	5.0	93.1	97.5	6.7	82.3	31.3	0.2	87.5	90.2	28.5	52.3	73.3
RBF	3.6	0.4	74.3	90.1	0.5	99.0	59.6	0.07	99.3	56.7	0.5	68.0	94.1	16.6	67.2	78.6
SVM	63.7	1.5	90.3	90.2	12.3	86.2	88.8	9.4	75.4	0	0	NaN	88.7	4.5	87.9	80.2
LIBSVM	70.2	4.9	92.3	92.3	7.4	88.2	87.2	10.3	70.9	10.9	0.8	71.8	89.7	6.3	90.7	87.4
KELM	2.3	52.4	89.5	54.2	1.4	96.7	91.5	7.2	80.2	29.3	0.14	81.2	96.2	20.5	65.2	79.9
CNN	71.9	10.4	70.9	91.8	4.5	95.5	97.2	2.2	92.6	50.7	0.5	93.2	97.7	5.8	84.4	89.4
DBN	62.4	14.3	60.3	89.6	5.7	93.5	94.4	2.8	89.7	54.2	0.11	90.2	95.9	9.7	77.7	86.5
DBN-KELM	64.2	2.1	88.9	89.9	2.3	96.3	93.8	3.3	89.6	82.8	0.4	92.3	94.3	7.2	80.2	88.7
DBN-EGWO-KELM	97.2	3.5	85.7	98.0	0.06	99.2	99.9	0.55	99.4	92.1	0.09	82.9	89.2	0.3	99.0	96.5
NSL-KDD																
Unit(%)	TPR	FPR	P	TPR	FPR	P	TPR	FPR	P	TPR	FPR	P	TPR	FPR	P	Acc
BP	3.3	0.6	75.6	80.2	7.5	92.4	96.6	12.3	82.6	31.2	1.3	88.2	87.2	30.9	53.3	73.0
RBF	4.2	1.1	72.1	88.7	1.9	97.7	60.3	3.1	98.4	55.5	2.2	59.6	84.6	40.2	67.9	77.1
SVM	50.3	2.6	88.2	86.9	19.4	86.6	87.2	13.9	74.3	0	0	NaN	90.3	6.9	85.6	80.2
LIBSVM	58.3	3.3	90.8	88.7	9.3	89.7	90.8	15.4	80.9	55.2	2.2	69.5	88.1	17.5	78.5	81.6
KELM	1.2	54.3	85.6	56.6	6.2	96.8	88.2	8.9	79.5	28.3	3.0	70.6	95.6	30.2	64.3	76.8
CNN	70.2	10.8	62.7	91.5	3.3	95.4	89.4	1.5	89.4	63.4	2.6	85.9	95.7	14.2	79.4	88.4
DBN	60.4	16.7	55.3	90.2	6.7	92.1	89.9	3.2	85.3	49.3	1.3	82.3	94.2	13.5	75.4	87.1
DBN-KELM	65.3	3.5	80.9	85.6	3.8	97.2	93.6	4.2	89.1	81.5	0.9	86.6	95.6	7.3	78.9	89.2
DBN-EGWO-KELM	92.2	4.9	78.7	96.5	0.5	98.2	98.1	1.1	98.1	90.2	0.2	86.5	87.5	1.1	96.5	94.0
	Normal			Fuzzers			Analysis			Backdoors			Dos			
UNSW-NB15																
Unit(%)	TPR	FPR	P	TPR	FPR	P	TPR	FPR	P	TPR	FPR	P	TPR	FPR	P	
BP	52.7	2.5	72.9	27.6	6.8	55.4	44.7	56.6	33.9	0.0	0.0	20.9	10.6	24.0	19.6	
RBF	56.6	49.3	83.7	14.6	11.2	60.3	3.5	25.4	25.9	0.8	0.8	25.3	23.5	4.8	22.6	
SVM	89.2	28.9	76.9	55.7	9.5	59.2	55.5	36.8	44.8	27.6	3.2	18.9	30.9	2.5	34.6	
LIBSVM	82.6	30.5	80.7	33.9	25.3	63.9	69.4	23.7	55.7	22.1	2.9	20.4	44.2	5.4	30.9	
KELM	81.6	14.9	77.5	69.6	17.6	60.9	1.8	44.8	63.2	1.7	2.3	0.9	35.6	0.0	12.3	
CNN	88.6	20.4	81.2	59.6	15.7	65.2	24.9	68.4	30.1	0.0	1.9	10.3	33.4	0.8	29.1	
DBN	87.9	23.5	78.9	54.3	13.3	64.4	7.6	69.2	25.8	0.0	0.0	0.5	40.2	0.0	25.8	
DBN-KELM	91.5	26.4	89.0	55.6	22.3	65.2	29.6	41.5	36.8	0.0	0.0	0.0	39.5	0.0	30.9	
DBN-EGWO-KELM	92.4	6.2	89.3	62.8	4.7	65.4	64.8	1.3	10.0	0.0	1.2	0.0	41.4	7.6	7.6	
	Exploits			Generic			Reconnaissance			Shellcode			Worms			
Unit(%)	TPR	FPR	P	TPR	FPR	P	TPR	FPR	P	TPR	FPR	P	TPR	FPR	P	Acc
BP	45.6	0.0	78.9	50.9	1.9	80.7	0.0	0.0	NaN	44.5	5.3	3.3	12.9	5.9	0.03	50.9
RBF	46.8	33.8	75.6	61.6	8.4	78.8	0.0	0.0	NaN	23.6	8.3	4.2	25.6	30.6	0.2	45.6
SVM	55.6	27.1	82.7	70.9	2.9	81.1	33.8	5.3	55.6	50.3	3.9	6.9	33.5	21.3	0.9	52.9
LIBSVM	60.3	15.4	83.2	80.4	5.7	86.9	48.6	9.4	67.2	55.4	5.9	3.7	40.7	20.1	2.9	55.4
KELM	42.2	25.0	80.3	57.7	4.9	88.5	57.9	3.7	62.3	49.6	3.7	5.5	40.6	15.9	1.3	65.9
CNN	62.1	22.7	89.4	79.4	7.7	89.7	54.1	5.6	52.7	59.2	2.9	6.4	60.7	12.3	3.7	70.2
DBN	56.9	3.5	85.6	72.6	17.6	90.2	25.0	8.8	66.9	57.3	1.3	7.9	45.9	3.9	2.2	68.6
DBN-KELM	59.6	4.5	86.9	80.2	15.5	92.3	60.9	3.9	67.6	58.9	0.9	11.3	48.3	1.2	2.5	75.6
DBN-EGWO-KELM	60.4	2.9	90.1	99.8	0.9	97.7	66.7	2.4	68.0	62.0	0.68	15.2	50.0	0.08	3.2	79.5

TABLE 11. *(Continued.)* **Experimental results of multi classification of each algorithm on test set.**

DBN	55.6	3.6	84.5	77.2	17.7	91.2	24.6	6.8	66.8	56.2	2.2	12.7	30.9	7.9	3.4	69.2
DBN-KELM	59.4	5.4	85.9	82.9	15.9	92.5	58.2	4.0	67.0	57.4	1.9	13.9	45.3	3.3	5.3	74.4
DBN-EGWO-KELM	60.1	3.1	89.6	99.8	0.8	98.0	64.8	2.6	67.1	59.3	0.69	15.0	NaN	0.1	0.0	79.4
	Normal			SSH-Patator			FTP-Patator			Dos			Web			
CICIDS2017																
Unit(%)	TPR	FPR	P	TPR	FPR	P	TPR	FPR	P	TPR	FPR	P	TPR	FPR	P	
BP	71.4	10.2	89.5	92.5	21.3	85.2	36.6	1.6	81.5	83.6	26.6	62.3	73.3	83.4	11.1	
RBF	77.7	9.2	92.5	61.6	11.9	92.5	54.6	6.2	62.2	86.7	31.9	66.7	74.6	84.7	13.2	
SVM	85.7	22.3	83.4	84.7	12.4	72.3	61.3	6.1	70.3	87.5	20.3	74.1	82.2	86.6	20.3	
LIBSVM	86.4	3.8	84.3	85.5	22.3	81.3	76.2	2.2	74.9	89.3	27.9	88.2	83.3	83.4	13.6	
KELM	63.3	9.6	92.7	88.4	5.8	81.2	32.9	6.9	71.6	90.2	22.3	68.3	77.2	62.3	15.6	
CNN	79.1	8.4	90.6	87.1	6.7	84.4	41.9	10.5	77.4	86.5	14.6	74.6	83.2	74.9	17.4	
DBN	87.2	5.8	91.2	90.5	9.6	81.6	46.0	2.9	822	92.8	13.3	81.9	87.6	87.2	15.3	
DBN-KELM	86.4	1.5	94.2	94.7	7.5	87.5	1.6	1.9	84.6	92.9	19.6	85.7	86.3	86.3	6.5	
DBN-EGWO-KELM	92.1	0.5	98.7	97.3	1.3	94.6	77.1	0.32	72.4	89.4	1.4	92.9	94.5	92.3	0.34	
	Bot			DDoS			PortScan									
Unit(%)	TPR	FPR	P	TPR	FPR	P	TPR	FPR	P							Acc
BP	85.6	1.8	75.9	82.5	10.8	84.2	31.3	0.2	87.5							63.5
RBF	84.7	6.2	73.8	60.2	9.4	81.4	56.7	0.5	68.0							62.7
SVM	88.5	5.1	89.3	75.1	15.6	75.6	0	0	NaN							79.3
LIBSVM	89.4	2.2	91.1	76.9	7.8	68.2	10.9	0.8	71.8							81.6
KELM	94.5	6.9	86.5	75.3	5.6	72.6	29.3	0.14	81.2							79.4
CNN	84.5	11.5	84.3	72.4	16.4	78.3	50.7	0.5	93.2							77.2
DBN	92.6	2.9	78.3	80.2	17.7	67.2	54.2	0.11	90.2							83.2
DBN-KELM	94.9	3.9	80.9	84.4	19.9	60.4	82.8	0.4	92.3							84.4
DBN-EGWO-KELM	87.4	0.32	83.6	70.4	0.8	80.9	92.1	0.09	82.9							97.1

data features by employing the dimensionality reduction ability of DBN for complex high-dimensional network intrusion data features. The combination with the enhanced grey wolf optimizer is viable to optimize the kernel extreme learning machine classification model for the purpose of improving the performance of KELM. The DBN-EGWO-KELM model, on the one hand, uses RBM to ensure the feature extraction performance of DBN; on the other hand, it uses the optimization capability of EGWO to enable KELM to quickly obtain the optimal E, s and other parameters, thereby enhancing KELM's high-dimensional Data classification capabilities. The experimental results show that: 1) The data classification performance of the DBN-EGWO-KELM intrusion detection model is stable and not sensitive to specific datasets; 2) The DBN-EGWO-KELM intrusion detection model successfully solves the problems of low accuracy, precision and true positive rate of existing methods; 3) On the four network intrusion detection datasets, the DBN-EGWO-KELM intrusion detection model has obvious advantages in various evaluation indicators compared with existing methods. The DBN-EGWO-KELM intrusion detection model provides a new and feasible solution for network security detection. In the future, we will continue to conduct in-depth research on the dimensionality reduction and classification of deep belief networks, and expand the application scope of the model.

REFERENCES

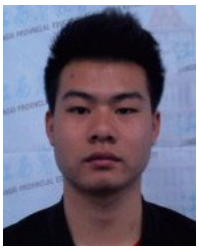
[1] S. F. Jilani, Q. H. Abbasi, and A. Alomainy, "Inkjet-printed millimetre-wave PET-based flexible antenna for 5G wireless applications," in *IEEE MTT-S Int. Microw. Symp. Dig.*, Aug. 2018, pp. 1–3, doi: 10.1109/IMWS-5G.2018.8484603.

[2] J. Shen, T. Zhou, D. He, Y. Zhang, X. Sun, and Y. Xiang, "Block design-based key agreement for group data sharing in cloud computing," *IEEE Trans. Depend. Sec. Comput.*, vol. 16, no. 6, pp. 996–1010, Nov./Dec. 2019, doi: 10.1109/TDSC.2017.2725953.

[3] C. Lee and A. Fumagalli, "Internet of Things security–Multilayered method for end to end data communications over cellular networks," in *Proc. IEEE 5th World Forum Internet Things (WF-IoT)*, Apr. 2019, pp. 24–28, doi: 10.1109/WF-IoT.2019.8767227.
[4] M. Kumar and A. K. Singh, "Distributed intrusion detection system using blockchain and cloud computing infrastructure," in *Proc. 4th Int. Conf. Trends Electron. Informat. (ICOEI)(4)*, Jun. 2020, pp. 248–252, doi: 10.1109/ICOEI48184.2020.9142954.
[5] F. Salo, M. Injadat, A. B. Nassif, A. Shami, and A. Essex, "Data mining techniques in intrusion detection systems: A systematic literature review," *IEEE Access*, vol. 6, pp. 56046–56058, 2018, doi: 10.1109/ACCESS.2018.2872784.
[6] W. Wang, X. Du, D. Shan, R. Qin, and N. Wang, "Cloud intrusion detection method based on stacked contractive auto-encoder and support vector machine," *IEEE Trans. Cloud Comput.*, early access, Jun. 9, 2020, doi: 10.1109/TCC.2020.3001017.
[7] X. Gao, C. Shan, C. Hu, Z. Niu, and Z. Liu, "An adaptive ensemble machine learning model for intrusion detection," *IEEE Access*, vol. 7, pp. 82512–82521, 2019, doi: 10.1109/ACCESS.2019.2923640.
[8] I. Ahmad, M. Basheri, M. J. Iqbal, and A. Rahim, "Performance comparison of support vector machine, random forest, and extreme learning machine for intrusion detection," *IEEE Access*, vol. 6, pp. 33789–33795, 2018, doi: 10.1109/ACCESS.2018.2841987.
[9] B. Naik, M. S. Obaidat, J. Nayak, D. Pelusi, P. Vijayakumar, and S. H. Islam, "Intelligent secure ecosystem based on Metaheuristic and functional link neural network for edge of things," *IEEE Trans. Ind. Informat.*, vol. 16, no. 3, pp. 1947–1956, Mar. 2020, doi: 10.1109/TII.2019.2920831.
[10] G. E. Hinton, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, Jul. 2006.
[11] Z. Yan and Y. Xu, "A multi-agent deep reinforcement learning method for cooperative load frequency control of a multi-area power system," *IEEE Trans. Power Syst.*, vol. 35, no. 6, pp. 4599–4608, Nov. 2020, doi: 10.1109/TPWRS.2020.2999890.
[12] D. W. Otter, J. R. Medina, and J. K. Kalita, "A survey of the usages of deep learning for natural language processing," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, Apr. 21, 2020, doi: 10.1109/TNNLS.2020.2979670.
[13] C. Li, J. Wang, H. Wang, M. Zhao, W. Li, and X. Deng, "Visual-texual emotion analysis with deep coupled video and danmu neural networks," *IEEE Trans. Multimedia*, vol. 22, no. 6, pp. 1634–1646, Jun. 2020, doi: 10.1109/TMM.2019.2946477.
[14] F. A. Khan, A. Gumaei, A. Derhab, and A. Hussain, "A novel two-stage deep learning model for efficient network intrusion detection," *IEEE Access*, vol. 7, pp. 30373–30385, 2019, doi: 10.1109/ACCESS.2019.2899721.
[15] L. Nie, Z. Ning, X. Wang, X. Hu, J. Cheng, and Y. Li, "Data-driven intrusion detection for intelligent Internet of vehicles: A deep convolutional neural network-based method," *IEEE Trans. Netw. Sci. Eng.*, vol. 7, no. 4, pp. 2219–2230, Oct. 2020, doi: 10.1109/TNSE.2020.2990984.
[16] T. Su, H. Sun, J. Zhu, S. Wang, and Y. Li, "BAT: Deep learning methods on network intrusion detection using NSL-KDD dataset," *IEEE Access*, vol. 8, pp. 29575–29585, 2020, doi: 10.1109/ACCESS.2020.2972627.
[17] K. Zhu, Z. Chen, Y. Peng, and L. Zhang, "Mobile edge assisted literal multi-dimensional anomaly detection of in-vehicle network using LSTM," *IEEE Trans. Veh. Technol.*, vol. 68, no. 5, pp. 4275–4284, May 2019, doi: 10.1109/TVT.2019.2907269.
[18] Q. M. Alzubi, M. Anbar, Z. N. M. Alqattan, M. A. Al-Betar, and R. Abdullah, "Intrusion detection system based on a modified binary grey wolf optimisation," *Neural Comput. Appl.*, vol. 32, no. 10, pp. 6125–6137, May 2020.
[19] B. Riyaz and S. Ganapathy, "A deep learning approach for effective intrusion detection in wireless networks using CNN," *Soft Comput.*, vol. 24, no. 22, pp. 17265–17278, Nov. 2020.
[20] P. Mishra, V. Varadharajan, U. Tupakula, and E. S. Pilli, "A detailed investigation and analysis of using machine learning techniques for intrusion detection," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 1, pp. 686–728, 1st Quart., 2019, doi: 10.1109/comst.2018.2847722.
[21] T. Treebupachatsakul and S. Poomrittigul, "Bacteria classification using image processing and deep learning," in *Proc. 34th Int. Tech. Conf. Circuits/Syst., Comput. Commun. (ITC-CSCC)*, Jun. 2019, pp. 1–3, doi: 10.1109/ITC-CSCC.2019.8793320.
[22] K. Nugroho, E. Noersasongko, Purwanto, Muljono, and H. A. Santoso, "Javanese gender speech recognition using deep learning and singular value decomposition," in *Proc. Int. Seminar Appl. Technol. Inf. Commun. (iSemantic)*, Sep. 2019, pp. 251–254, doi: 10.1109/ISEMANTIC.2019.8884267.
[23] A. R. Sharma and P. Kaushik, "Literature survey of statistical, deep and reinforcement learning in natural language processing," in *Proc. Int. Conf. Comput., Commun. Autom. (ICCCA)*, May 2017, pp. 350–354, doi: 10.1109/CCAA.2017.8229841.
[24] Y. Xin, L. Kong, Z. Liu, Y. Chen, Y. Li, H. Zhu, M. Gao, H. Hou, and C. Wang, "Machine learning and deep learning methods for cybersecurity," *IEEE Access*, vol. 6, pp. 35365–35381, 2018.
[25] W. Lee and S. J. Stolfo, "A framework for constructing features and models for intrusion detection systems," *ACM Trans. Inf. Syst. Secur.*, vol. 3, no. 4, pp. 227–261, Nov. 2000, doi: 10.1145/382912.382914.
[26] L. Ertöz, M. Steinbach, and V. Kumar, "Finding clusters of different sizes, shapes, and densities in noisy, high dimensional data," in *Proc. SIAM Int. Conf. Data Mining*, May 2003, pp. 47–58.
[27] N. B. Amor, S. Benferhat, and Z. Elouedi, "Naive Bayesian networks in intrusion detection systems," in *Proc. 23rd Workshop Probabilistic Graph. Models Classi_cation, 14th Eur. Conf. Mach. Learn. (ECML) 7th Eur. Conf. Princ. Pract. Knowl. Discovery Databases (PKDD)*, Cavtat-Dubrovnik, Croatia, 2003, p. 11.
[28] W. Li, "Using genetic algorithm for network intrusion detection," in *Proc. United States Dept. Energy Cyber Secur. Group Training Conf.*, 2004, pp. 24–27.
[29] L. Didaci, G. Giacinto, and F. Roli, "Ensemble learning for intrusion detection in computer networks," in *Proc. Workshop Mach. Learn. Methods Appl.*, Siena, Italy, 2002, pp. 1–11.
[30] C. Kolias, G. Kambourakis, and M. Maragoudakis, "Swarm intelligence in intrusion detection: A survey," *Comput. Secur.*, vol. 30, no. 8, pp. 625–642, Nov. 2011, doi: 10.1016/j.cose.2011.08.009.
[31] R. C. Staudemeyer, "Applying long short-term memory recurrent neural networks to intrusion detection," *South Afr. Comput. J.*, vol. 56, pp. 136–154, Jul. 2015.
[32] C. Yin, Y. Zhu, J. Fei, and X. He, "A deep learning approach for intrusion detection using recurrent neural networks," *IEEE Access*, vol. 5, pp. 21954–21961, 2017.
[33] J. Kim, J. Kim, H. L. T. Thu, and H. Kim, "Long short term memory recurrent neural network classi_er for intrusion detection," in *Proc.Int. Conf. Platform Technol. Service (PlatCon)*, Feb. 2016, pp. 1–5.
[34] G. E. Hinton and R. R. Salakhutdinov, "Supporting online material for 'reducing the dimensionality of data with neural networks,'" *Science*, vol. 28, 2006, Art. no. 313504.
[35] L. Yong, W. Min, C. Weihua, L. Xuzhi, and W. Chunsheng, "PSO-BP control algorithm based on particle size distribution evaluation and optimization," *J. Automat.*, vol. 38, no. 6, pp. 1007–1016, 2012.
[36] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, "Extreme learning machine: A new learning scheme of feedforward neural networks," in *Proc. IEEE Int. Joint Conf. Neural Netw.*, Jul. 2004, pp. 985–990, doi: 10.1109/IJCNN.2004.1380068.
[37] H.-T. Li, C.-Y. Chou, Y.-T. Chen, S.-H. Wang, and A.-Y. Wu, "Robust and lightweight ensemble extreme learning machine engine based on eigenspace domain for compressed learning," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 66, no. 12, pp. 4699–4712, Dec. 2019, doi: 10.1109/TCSI.2019.2940642.
[38] N. D. Vanli, M. O. Sayin, I. Delibalta, and S. S. Kozat, "Sequential nonlinear learning for distributed multiagent systems via extreme learning machines," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 3, pp. 546–558, Mar. 2017, doi: 10.1109/TNNLS.2016.2536649.
[39] S. Jain, M. Singhal, and S. Shukla, "Comments on 'traffic sign recognition using kernel extreme learning machines with deep perceptual Features,'" *IEEE Trans. Intell. Transp. Syst.*, vol. 20, no. 10, pp. 3759–3761, Oct. 2019, doi: 10.1109/TITS.2018.2850057.
[40] X. Jiaming, Z. Weiqiang, Y. Dengzhou, L. Jia, and X. Shanhong, "Language recognition system based on Manifold Regularization limit learning machine," *J. Automat.*, vol. 41, no. 9, pp. 1680–1685, 2015.
[41] G. B. Huang, "An insight into extreme learning machines: Random neurons, random features and Kernels," *Cogn. Comput.*, vol. 6, no. 3, pp. 1–15, 2014.
[42] S. Mirjalili, S. M. Mirjalili, and A. Lewis, "Grey wolf optimizer," *Adv. Eng. Softw.*, vol. 69, pp. 46–61, Mar. 2014.

[43] C. Li, B. Feng, S. Li, J. Kurths, and G. Chen, "Dynamic analysis of digital chaotic maps via state-mapping networks," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 66, no. 6, pp. 2322–2335, Jun. 2019, doi: 10.1109/TCSI.2018.2888688.
[44] C. Alippi and M. Roveri, "Virtual k-fold cross validation: An effective method for accuracy assessment," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2010, pp. 1–6.
[45] A. Chalimourda, B. Scholkopf, and A. Smola, "Experimental optimal V in support vector regression for different noise models and parameter settings," *Neural Netw.*, vol. 17, no. 1, pp. 127–141, 204.
[46] X. Li and K. M. Luk, "The grey wolf optimizer and its applications in electromagnetics," *IEEE Trans. Antennas Propag.*, vol. 68, no. 3, pp. 2186–2197, Mar. 2020, doi: 10.1109/TAP.2019.2938703.
[47] S. Choudhary and N. Kesswani, "Analysis of KDD-Cup'99, NSL-KDD and UNSW-NB15 datasets using deep learning in IoT," *Procedia Comput. Sci.*, vol. 167, pp. 1561–1573, Jan. 2020.
[48] N. Shone, T. N. Ngoc, V. D. Phai, and Q. Shi, "A deep learning approach to network intrusion detection," *IEEE Trans. Emerg. Topics Comput. Intell.*, vol. 2, no. 1, pp. 41–50, Feb. 2018, doi: 10.1109/TETCI.2017.2772792.
[49] B. Roy and H. Cheung, "A deep learning approach for intrusion detection in Internet of Things using bi-directional long short-term memory recurrent neural network," in *Proc. 28th Int. Telecommun. Netw. Appl. Conf. (ITNAC)*, Nov. 2018, pp. 1–6.
[50] Z. Wang, *The Applications of Deep Learning on Traffic Identification*. San Francisco, CA, USA: BlackHat, 2015, pp. 21–26.

ZHENDONG WANG (Member, IEEE) received the B.E. degree from the Changchun University of Science and Technology, in 2006, the M.Eng. degree from the Harbin University of Science and Technology, in 2009, and the Ph.D. degree in computer applied technology from the Harbin Engineering University, in 2013. Since 2014, he has been with the Department of Information Engineering, Jiangxi University of Science and Technology, China, where he is currently an Associate Professor. His research interests include wireless sensor networks, artificial intelligence, and network security.



YONG ZENG received the B.S. degree in network engineering from the Jiangxi University of Science and Technology, in 2018, where he is currently pursuing the M.S. degree. he main research interests include network security and group intelligence optimization algorithms.

YAODI LIU received the B.S. degree in information and computing science from Jiangsu Ocean University, in 2018. She is currently pursuing the M.S. degree. Her main research interests include network security and group intelligence optimization algorithms.

DAHAI LI received the Ph.D. degree. He is currently an Associate Professor with the School of Information Engineering, Jiangxi University of Science and Technology. His main research interests include distributed system quality of service (QoS) control and distributed system self-learning resource scheduling control.