EAI Endorsed Transactions
on Scalable Information Systems

Research Article EAI.EU

A Bagging Strategy-Based Kernel Extreme Learning Machine for Complex Network Intrusion Detection

Shoulin Yin[1], Hang Li[1,*], Asif Ali Laghari[2], Shahid Karim[3], Awais Khan Jumani[3]

[1]Software College, Shenyang Normal University, Shenyang 110034 China
[2]Department of Computer Science, Sindh Madressatul Islam University, Karachi, Pakistan
[3]Department of Computer Science, ILMA University, Karachi, Pakistan

Abstract

Network intrusion can enter the network through informal channels. Some illegal users utilize Trojans and self-programmed attack to change the network security system, so that the system loses the defense and alarm function and the Hacker can steal the internal information. Network intrusion seriously harms the security of network information and the legitimate rights of users. Therefore, a bagging strategy-based kernel extreme learning machine for complex network intrusion detection is presented in this paper. This method adopts a bagging strategy to train several sub-kernel extreme learning machines independently. Then the integrated gain of above machines is measured based on the margin distance minimization (MDM) criterion. Selected machines with high gain degree are selected for selective integration to obtain selective integrated learners with strong generalization ability and high efficiency. Then an improved universal gravitation search algorithm is used to optimize the kernel parameters. Meanwhile, a sub-kernel extreme learning machine online update strategy based on incremental learning of batch samples is introduced to realize the online update of intrusion detection model, so that the proposed detection method can effectively be adapted to the changes of complex network environment. Finally, experiments illustrate that the proposed method has better effect on network intrusion detection in terms of detection accuracy and speed, especially for unknown network intrusion connection events, the response speed is fast, the false alarm rate is low.

Keywords: network intrusion detection, bagging strategy, kernel extreme learning machine, margin distance minimization, online update strategy.

Received on 28 July 2021, accepted on 06 September 2021, published on 06 October 2021

doi: 10.4108/eai.6-10-2021.171247

*Corresponding author. Email: lihangsoft@163.com

1. Introduction

With the arrival of the era of big data, network information is facing more and more security threats. As an active security protection technology, network intrusion detection is expected to intercept and respond to intrusion before harming the network system, which has received extensive attention from many researchers [1-3].

Traditional intrusion detection [4-6] includes two detection types, namely, misuse detection and anomaly detection. Misuse detection finds the abnormal links in the network by establishing an intrusion rule base. Although the accuracy is high, but it is powerless for the new intrusion types and the old virus variant connection. Anomaly detection is used to analyse network anomalies by summarizing the characteristics of normal network connections. Because this method also has a good detection effect against the new attacks, it has been widely concerned.

Network intrusion detection based on automatic learning is to transform network intrusion detection into pattern recognition (classification). These methods train the classifier (including SVM, ELM, DMT) to identify the normal behavior and abnormal behavior in the network connection [7-9]. Generally speaking, the training and testing time of complex classifier is relatively long. Although the simple classifier has high processing efficiency, it may be difficult to obtain effective recognition effect for network attack events with multiple features and complex intrusion modes. Some researchers hope to integrate the advantages of multiple learners to obtain better detection performance. Therefore, in recent years, intrusion detection methods based on integrated hybrid learning approaches have attracted wide attention.

The integration learning algorithm improves the generalization ability of the whole algorithm by integrating multiple sub-learners [10]. In theory, intrusion detection based on integrated learning is much better than intrusion detection based on single learner. However, not every sub-learner is beneficial for integrated learning. If we can select the better learning part for selective integration, the integrated learning will have better performance and can improve the detection efficiency.

In this paper, in the complex network environment, the network connection is complex and changeable [11, 12]. Cyber-attacks emerge in an endless stream. The existing network intrusion detection methods have problems such as slow recognition speed and low recognition rate for new intrusion modes. As we all know, the kernel extreme learning machine (KELM) has the advantage of fast learning, then we combine KELM and bagging learning strategy, a network intrusion detection method based on KELM selective integrated learning, named BL-KELM is proposed. BL-KELM selectively integrates partial sub-learners by subdividing the gain of each KELM on the ensemble detector. Then we use improved universal gravitation algorithm (UGA). Meanwhile, this new method can update the model online according to the change of network environment to enhance the detection efficiency of known intrusion types and decrease the false alarm rate of unknown intrusion types. Finally, we make experiments on the traditional KDD99 data set and a manually built complex network physical simulation platform to verify the effectiveness of the proposed BL-KELM method.

2. Proposed network intrusion detection

Network intrusion detection is essentially a multi-variable pattern classification problem. Suppose that n network connection data sets $X = \{X_i \mid i = 1,2,\cdots,n\}^T \in R^{n\times K}$ are collected. Where $X_i \in R^K$ represents the i-th network connection data record. K Represents the feature dimension of the network connection record data. n is the collected sample number. The network connection type corresponding to these records is marked as $T = \{T_i \mid i = 1,2,\cdots,n\}^T$. Then the intrusion detection model based on single learner is:

$$G_j = \min_{G_j}\{\| T - G_j(x_i) \|_2^2 + \lambda\varphi(G_j)\} \qquad (1)$$

Where $\varphi(\cdot)$ is the regular term to control the complexity of the model. λ is the corresponding weight coefficient. If the structure of G_j has been determined, then formula (1) is equivalent to obtaining an optimal classifier by adjusting the parameters in classifier G_j . The proposed BL-KELM method in this paper is based on the bagging learning strategy. Firstly, a group of sub-learners with certain complementary functions are trained in parallel, and then MDMC is used to selectively learn the sub-learners. The flow diagram of this proposed method is shown in figure 1.

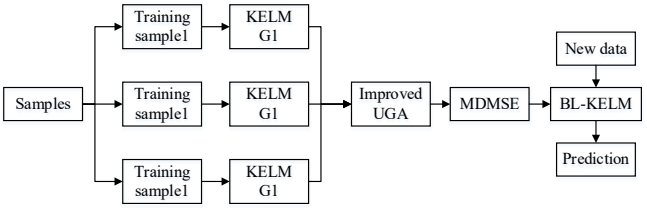

Figure 1. Proposed BL-KELM structure.

2.1. KELM sub-classifier

ELM is essentially a linear parametric equation. A closed-form theoretical global optimal solution can be obtained by solving this linear system. Suppose there are N network connection data samples $\{(X_i, T_i) \mid i = 1,2,\cdots,n\}$. Where $X_i \in R^n$ represents network connection data feature. $T_i \in R^m$ denotes the corresponding label of each network connection data record [13,14].

The output of ELM with ω hidden layer nodes can be written as:

$$O_j = \sum_{i=1}^{\omega} \beta_i h_i(X_j) = h(X_j)\beta, j = 1,2,\cdots,m \quad (2)$$

Where $\beta = (\beta_1,\cdots,\beta_\omega)^T \in R^{\omega \times m}$ is the weight vector of the output layer. $h(X) = (h_1(x),\cdots,h_\omega(X))$ is a nonlinear mapping of ELM. $h_i(X)$ is the output of the i-th hidden neuron.

$$h_i(X) = g(W_i, b_i, X) \quad (3)$$

Where $g(\cdot)$ denotes the activation function of the hidden layer. The commonly used activation functions include sigmoid function, hyperbolic tangent function, radial basis function, etc. W_i and b_i represent the input weight and bias of the i-th hidden layer respectively. The input weight W_i and bias b_i are randomly initialized without iterative modification. The basic ELM learner obtains output layer parameters by solving the following linear optimization problems.

$$\beta_{BELM} = \min_{\beta} \| H\beta - T \|_F^2 \quad (4)$$

F stands for Frobenius norm. H is the output matrix of the hidden layer.

$$H = \begin{bmatrix} h(X_1) \\ \vdots \\ h(X_N) \end{bmatrix} = \begin{bmatrix} h_1(X_1)\cdots h_L(X_1) \\ \vdots \quad \ddots \quad \vdots \\ h_1(X_N)\cdots h_L(X_N) \end{bmatrix} \quad (5)$$

$T = [T_1, T_2, \cdots, T_N]^T$ is the target (label) matrix of the training data set. It can be obtained by solving the optimization problem (4).

$$\beta_{BELM} = H^+ T \quad (6)$$

Where H^+ denotes the Moore-Penrose generalized inverse matrix of H. In order to improve the generalization performance of ELM, a regular term is usually added when the approximation error is considered to obtain better performance. Based on this principle, formula (4) can be adjusted to a general ELM model with regular terms [15].

$$\beta_{RELM} = \min_{\beta} \{ \| H\beta - T \|_q^{\sigma_1} + \lambda \| \beta \|_p^{\sigma_2} \} \quad (7)$$

Where $\sigma_1 > 0, \sigma_2 > 0$, $p, q \in [0,0.5,1,2,\cdots,\infty]$ is the norm. When $\sigma_1 = \sigma_2 = 2$, p=2, q=F, and training sample number N is greater than feature number L.

$$\beta_{RELM} = H^T (I/\lambda + HH^T)^{-1} T \quad (8)$$

After obtaining the model parameter β_{RELM} of ELM classifier, for the multi-category network intrusion detection problem, if given a test sample X_{test}, its corresponding network intrusion type j is the network intrusion type corresponding to the position of the $h(X_{test})\beta$ vector maximum value, namely,

$$j = \max_j \{ h(X_{test})\beta \} \quad (9)$$

The parameter vector of ELM output layer described in formula (6) or (8) is directly related to the feature mapping function of the hidden layer. If it can avoid the influence of hidden layer feature mapping function on ELM model, then KELM model is a better choice.

According to the calculation results of β_{RELM}, given a new network connection record X_{test}, the corresponding connection type vector can be expressed as:

$$y_{test} = h(X_{test})\beta_{RELM} = h(X_{test})H^T (I/\lambda + HH^T)^{-1} T \quad (10)$$

Where HH^T can be represented by the ELM kernel matrix.

Let $\Omega = HH^T$ represent the ELM kernel matrix. Where $\Omega_{ij} = h(X_i)h(X_h)^T$ can be represented by the kernel function $K(X_i, X_j)$. The kernel matrix is substituted into equation (10), then

$$y_{test} = h(X_{test})(h^T(X_1),\cdots,h^T(X_N))^T (I/\lambda + \Omega)^{-1} T$$
$$= \begin{pmatrix} K(X_{test}, X_1) \\ \vdots \\ K(X_{test}, X) \end{pmatrix} \left(\frac{I}{\lambda} + \underbrace{\begin{pmatrix} K(X_1, X_1)\cdots K(X_1, X_N) \\ \vdots \quad \cdots \quad \vdots \\ K(X_N, X_1)\cdots K(X_N, X_N) \end{pmatrix}}_{\Omega} \right)^{-1} T \quad (11)$$

Formula (11) is the KELM expression formula, which does not need to explicitly provide the feature mapping function $h(\cdot)$ of the hidden layer and give the neuron number L of the hidden layer.

2.2. Improved UGA for optimizing KELM

The universal gravitation algorithm has the advantages of strong global optimization ability and simple process [16,17]. It can be well used in practical problems and can highlight its superiority in the treatment of nonlinear problems. It has been successfully applied to many problems in engineering and other fields. However, as a member of intelligent algorithms, gravitation algorithm has the defects of falling into local extremum and premature convergence. In order to overcome the above shortcomings, Opposition Based Learning (OBL) [18,19] is used to initialize the initial population of GSA to make the distribution of the initial population more uniform. Tent chaos mapping is introduced to improve the diversity of the population and promote the exploration and development of GSA algorithm.

The steps of OBL generation of initial population are as follows: 1) Determine the initial population number as the random generation of initial population (i.e. the first group of candidate solutions), calculate the fitness value of each individual. The second group of candidate solutions and fitness values are obtained. 2) The two groups of candidate solutions are sorted according to the size of individual fitness value, and the solution space of the problem is formed by the former particle, that is, the initial population of GSA.

The steps of Tent chaotic mapping to generate chaotic sequence are as follows: 1) Normalize the optimal solution found so far to the interval of (0,1), and make it equal to the initial value x_0, that is, k=0. 2) Chaotic sequence X is generated by iteration, and k++; 3) Stop iteration and save X sequence when the maximum number of iterations is reached; 4) The sequence X is inversely normalized into the original solution space to obtain a new solution. 5) Compare the fitness values of the new and old solutions, leaving the solution with better performance. The improved universal gravitation search algorithm is used to select the best kernel parameters and output weights intelligently. The KELM neural network is optimized to ensure the accuracy of model prediction.

2.3. KELM classifier online updating incrementally

Due to the complexity and variety of network intrusion models, it is difficult to adapt to the change of new network intrusion models, if it is only based on the classifier model obtained from the previous historical network connection data sets. Therefore, it is necessary to update the obtained classifier model (at any time/in real time).

Lim et al. [20] proposed online sequential ELM update algorithm based on recursive least squares (Online sequential ELM, OSELM). Inspired by this idea, we first train a set of KELM sub-learner models based on historical data sets. Then, in actual network security monitoring, each KELM sub-classifier model is updated online when a new set of samples with relatively explicit tags are collected (the sample tags can be based on manual tags or the result of a common decision based on multiple classifiers). Finally, the KELM sub-classifier models are then updated online.

The main steps of updating the KELM sub-learner model online are as follows:

1) First, the initial KELM classifier model $G^0(X)$ is obtained according to the initial training samples ($\prod_0 = \{X_i, T_i\}_{i=1}^{N_0}$, N_0 is the sample number. The feature map matrix of these initial samples is set as H_0, the corresponding label matrix is T_0). According to formula (11), $G^0(x) = K_0\Omega_0^{-1}T_0$, in here, $K_0 = (K(X_1, x), K(X_2, x), \cdots, K(X_{N_0}, x))^T$, $\Omega_0 = I_0/\lambda + \Omega_{KELM}^0$, $\Omega_{KELM}^0 = H_0H_0^T$.

2) In the process of network intrusion detection, if the new samples $\Delta\prod_1 = \{X_i, T_i\}_{i=N_0+1}^{N_0+N_1}$ with clear labels are obtained, at this time, the large sample set $\prod_1 = \{\prod_0, \Delta\prod_1\}$ constituted by the initial sample set $\prod_0$ and the newly collected sample $\Delta\prod_1$ is considered comprehensively. The corresponding KELM classifier model $G^1(X)$ can be expressed as:

$$G^1(X) = K_1\Omega_1^{-1}T_1 = \begin{pmatrix} K_0 \\ \Delta K_1 \end{pmatrix}\left(\frac{I_1}{\lambda} + \begin{pmatrix} H_0 & \Delta H_{01} \\ \Delta H_{10} & \Delta H_1 \end{pmatrix}\begin{pmatrix} H_0^T & \Delta H_{10}^T \\ \Delta H_{01}^T & \Delta H_1^T \end{pmatrix}\right)^{-1} \cdot \begin{pmatrix} T_0 \\ \Delta T_1 \end{pmatrix} \quad (12)$$

$$= \begin{pmatrix} K_0 \\ \Delta K_1 \end{pmatrix}\begin{pmatrix} I_0/\lambda + \Omega_{KELM}^0 & \Delta\Omega_{KELM}^{01} \\ \Delta\Omega_{KELM}^{10} & \Delta I_1/\lambda + \Delta\Omega_{KELM}^1 \end{pmatrix}^{-1} \cdot \begin{pmatrix} T_0 \\ \Delta T_1 \end{pmatrix}$$

Where $\Delta K_1 = (K(X_{N_0}+1), \cdots, K(X_{N_0+N_1}, X))^T$ denotes the kernel mapping of X on new training sample. $\Delta\Omega_{KELM}^1 = \Delta H_1 H_1^T \Delta T_1 = (T_{N_0+1}, \cdots, T_{N_0+N_1})^T$, $\Delta\Omega_{KELM}^{01} = H_0\Delta H_1^T$, $\Delta\Omega_{KELM}^{10} = (\Delta\Omega_{KELM}^{01})^T = \Delta H_1 H_0^T$. Specifically,

$$\Delta\Omega_{KELM}^{01} = \begin{pmatrix} K(X_1, X_{N_0+1}) \cdots K(X_1, X_{N_0+N_1}) \\ \vdots \quad \ddots \quad \vdots \\ K(X_{N_0}, X_{N_0+1}) \cdots K(X_{N_0}, X_{N_0+N_1}) \end{pmatrix} \in R^{N_0 \times N_1}$$

$$\Delta\Omega_{KELM}^{10}=\begin{pmatrix}K(X_{N_0+1},X_1)\cdots K(X_{N_0+1},X_{N_0})\\ \vdots \quad \ddots \quad \vdots\\ K(X_{N_0+N_1},X_1)\cdots K(X_{N_0+N_1},X_{N_0})\end{pmatrix}\in R^{N_1\times N_0}$$

$$\Delta\Omega_{KELM}^{1}=\begin{pmatrix}K(X_{N_0+1},X_{N_0+1})\cdots K(X_{N_0+1},X_{N_0+N_1})\\ \vdots \quad \ddots \quad \vdots\\ K(X_{N_0+N_1},X_{N_0+1})\cdots K(X_{N_0+N_1},X_{N_0+N_1})\end{pmatrix}\in R^{N_1\times N_1}$$

If let $L_0=I_0/\lambda+\Omega_{KELM}^0$, $L_1=\Delta I_1/\lambda+\Delta\Omega_{KELM}^1$, according to

$$\begin{pmatrix}A & B\\ C & D\end{pmatrix}^{-1}=\begin{pmatrix}A^{-1}+A^{-1}BMCA^{-1} & -A^{-1}BM\\ -MCA^{-1} & M\end{pmatrix},$$

where $M=(D-CA^{-1}B)^{-1}$, so we can get

$$\Omega_1^{-1}=\begin{pmatrix}L_0 & \Delta\Omega_{KELM}^{01}\\ \Delta\Omega_{KELM}^{10} & L_1\end{pmatrix}^{-1}=\begin{pmatrix}L_0^{-1}+D_1R_{KELM}D_1^T & -D_1R_{KELM}\\ -R_{KELM}D_1^T & R_{KELM}\end{pmatrix} \quad (13)$$

Where $R_{KELM}=(L_1-\Delta\Omega_{KELM}^{10}L_0^{-1}\Delta\Omega_{KELM}^{01})^{-1}$, $D_1=L_0^{-1}\cdot\Delta\Omega_{KELM}^{01}$.

By substituting equation (13) into equation (12), the KELM model $G^1(X)$ can be incrementally updated.

$$\begin{aligned}G^1(X)&=K_1(X)\Omega_1^{-1}T_1\\&=K_0L_0^TT_0+K_0D_1R_{KELM}D_1^TT_0-\Delta K_1R_{KELM}D_1^TT\\&\quad-K_0D_1R_{KELM}\Delta T_1+\Delta K_1R_{KELM}\Delta T_1\\&=G^0(X)+(K_0D_1-\Delta K_1)R_{KELM}(D_1^TT_0-\Delta T_1)\end{aligned} \quad (14)$$

3) Formula (14) is the batch updating processing of KELM. Assuming that h times of model updating have been carried out previously, and we obtain the model parameter $G^k(X)$, so the k+1-th KELM classifier online batch sample incremental updating criterion is:

$$\begin{aligned}G^{k+1}(X)&=G^k(X)+\\&(K_kD_{k+1}-\Delta K_{k+1})R_{KELM}(D_{k+1}^TT_k-\Delta T_{k+1})\end{aligned} \quad (15)$$

Where $K_k=(K_1^T,\Delta K_1^T,\cdots,\Delta K_k^T)^T$, $D_{k+1}=L_0^{-1}\Delta\Omega_{KELM}^{kk+1}$, $\Delta T_{k+1}=(T_{1+\sum_{i=1}^{k}N_i},\cdots,T_{1+\sum_{i=1}^{k+1}N_i})$, $T_k=(T_0^T,\Delta T_1^T,\cdots,\Delta T_k^T)^T$.

In practice, as the number of batch updates increasing, the matrix (vector) in the second term of equation (15) will become larger. In order to ensure the effectiveness of calculation, a forgetting mechanism can be added to discard some samples with a long history. For example, the maximum sample size is set as $N_{\max}$. If $\sum_{i=1}^{j}N_i>N_{\max}$ in the j-th batch updating. Then according to the historical order of samples, d samples with long history are discarded to ensure $\sum_{i=1}^{j}N_i-d\le N_{\max}$, which effectively updates the KELM model.

2.4. KELM sub-learner selective integration based on MDM (MDMSE)

Ensemble learning integrates several weak classifiers to form a new strong classifier algorithm, which is also called meta-algorithm such as boosting and bagging. Boosting optimizes each newly created sub-learner by focusing on the samples that are misclassified by the existing classifier. Boosting method emphasizes the strong dependence between sub-learners including Adaboost, XGBOOST, GBDT, etc. [21, 22]. Bagging is Bootstrap Aggregation, where Bootstrap is a random re-sampling [23, 24]. It advocates that each sub-learner should be as independent of each other as possible. The distributed concurrent computing method which can be generated at the same time is used to establish. Therefore, this method can train a group of sub-learners independently and efficiently, and make each sub-learner have complementary abilities [25, 26, 27].

In view of the concurrent learning advantage of the Bagging strategy, it is adopted for sub-classifier learning in this paper. Meanwhile, to ensure that each kind of maximum anomaly intrusion behavior can be detected, this paper proposes a margin distance minimum selective integration (MDMSE) algorithm to order of all sub-learning gain. By selecting the partial learner with large gain degree as the final integration result, the negative effect of the weak learner on the final detection result is reduced. Selective integrated learning refers to the integration of individual learners with a large gender and strong generalization ability from all trained sub-learners to obtain better performance. The study shows that selective integration algorithm is superior to the signal Bagging or Boosting algorithm. In order to avoid local optimization and improve computational efficiency, a new selective integration algorithm MDMSE is proposed based on MDM principle. Based on the MDM criterion, the method calculates the gain degree of each sub-learner for the performance improvement of the integrated algorithm. The KELM sub-learner with a high gain degree is selected for partial integration to obtain a strong learner with high computational efficiency and strong generalization

ability. Here are some basic definitions to analyze the main principles of MDMSE.

Definition 1. Classifier eigenvectors C_t.

Given a labeled data set D with n elements. The eigenvector C_t of classifier G_t is an n-dimensional vector, its i-th term is:

$$C_t^i = 2 \oplus (h_t(x_i) == y_i) - 1, (x_i, y_i) \in Z \quad (16)$$

Where $h_t(x_i)$ is the discriminant result of classifier t on the i-th sample. y_i is the label of the i-th sample. The symbol $\oplus$ represents the product of two numbers. When the classifier t is classified correctly on the i-th sample of data set, $C_t^i = 1$, otherwise $C_t^i = -1$. The overall eigenvector is the sum of all the classifier eigenvectors. The average value of this eigenvector is:

$$\overline{C} = \frac{1}{T} \sum_{t=1}^{T} C_t \quad (17)$$

Definition 2. N dimensional space optimization point O.

If the i-th part of the average eigenvector $\overline{C}$ is positive, the whole classifier is classified correctly on the i-th sample. Therefore, if the eigenvectors of a partial ensemble classifier are in the first quadrant of the n-dimensional space (that is, all the components are positive), the classification is correct on data set D. The goal of this paper is to select an ensemble learner that is as small as possible but whose average eigenvector is as close as possible to a reference position in the first quadrant. It selects any target position p as the point O with the same component, that is:

$$O_i = p, with\ i = 1,2,\cdots,N\ and\ 0 < p < 1 \quad (18)$$

Definition 3. Iterative selection of sub-classifiers based on integrated gain. The classifier whose distance from the eigenvector $\overline{C}$ to the target point O decreases the most is with the largest integrated gain. Therefore, the classifier selected in the u-th iteration is:

$$S_u = \arg\min_k d(O, \frac{1}{U}(C_k + \sum_{t=1}^{u-1} C_t)), k \in E_T / S_{u-1} \quad (19)$$

Where $d(v,u)$ is the Euclidean distance between point v and point u. S can be regarded as the distance gain of the current sub-learner. If the S is smaller, the gain is greater. Where $0 < p < 1$. Theoretically, p should be small enough, so that simple examples (those are correctly classified by most sub-learners) can be quickly close to the p value. Then, this allows the algorithm gradually focus on the more difficult classify samples. By contrast, if p is close to 1, it will have a similar attraction for all instances throughout the selection process, which will reduce the effectiveness of the method. To sum up, the main steps of the proposed BL-KELM algorithm in this paper are as algorithm 1.

Algorithm 1. BL-KELM.

Step 1. Input data $D = \{(X_i, Y_i) \mid i = 1,2,\cdots,N\}$. Set sub-learner number as T, selective learning integration number as U. The final selective ensemble learning set ST is initially null, the optimal position of n-dimensional space is p.

Step 2. Based on Bagging mechanism, the data set D is sampled and divided into T+1 sub-data sets.

Step 3. T KELM sub-learners $\{G_t \mid t = 1,2,\cdots,T\}$ are trained with T data subsets.

Step 4. The gain of each sub-learner is calculated on the T+1 data set according to equation (19).

Step 5. It selects the sub-learner with the greatest gain and adds it to the set ST.

Step 6. Judging whether the number of learners in ST is greater than U. If the condition is satisfied, then it stops; otherwise, repeat Step5.

3. Experiment and analysis

The experiments mainly consist of two parts: (1) verifying the effectiveness of the proposed BL-KELM on the KDD99 data set, analysing the impact of different parameters on the performance and comparing the performance of BL-KELM with the state-of-the-art intrusion detection methods. (2) Testing the real-time performance and effectiveness of BL-KELM in the real intrusion detection types under a complex network environment.

3.1. KDD data set experimental results

KDD99 is a network connection data set simulated and collected by the advanced planning department of the U.S. defence department at MIT Lincoln laboratory. It contains about 5 million network connection data sets, which can be divided into training sets and tests set including 4 big categories and 39 small classes of abnormal intrusion types and normal connection. The training set contains 22 kinds of abnormal attack type, the remaining 17 as unknown type in the test set (KDD99 provides 10% test set) used for judging the unknown intrusion test, it will also be as an important basis to verify the robustness of the BL-KELM. Each connection and each label have 42-dimension features. In this experiment, the KDD99 data set is preprocessed first, the character features of the data set are converted into

numerical features, and the feature data is normalized. Then, the processed training data set is randomly sampled. T+1 sub-data sets are generated. The first T part is used for learner training, and the last T+1 part is used for selection of sub-learners. Considering the second manual examination for network intrusion and the serious harm caused by network intrusion, network intrusion detection should check out all abnormal connections as much as possible. In this paper, accuracy rate (AR) and missing rate (MR) are used as two evaluation indexes for evaluation of proposed method.

$$AR=(TP+TN)/(N+P) \quad (20)$$

$$MR=1-TN/(TN+FP) \quad (21)$$

Where P is the positive sample number and N is the negative sample number. TP is actually positive sample and classified as positive sample number. TN is the actually negative sample and classified as positive sample number. FP is the actually negative sample but classified as positive sample number. Accuracy rate can objectively evaluate the performance of an algorithm, while missing rate can effectively evaluate the generalization of the algorithm.

This experiment consists of three parts: (1) a verification experiment: verifying the effectiveness of BL-KELM, and comparing with single KELM and traditional integrated KELM; (2) influence of parameter: analysing the influence of different parameters on BL-KELM; (3) comparison experiment.

1) verification experiment

We compare the BL-KELM, KELM and KELM integration algorithm on KDD99 as shown in table 1. In the experiment, the radial basis function (RBF) is adopted as the kernel function of KELM. BL-KELM and KELM integration algorithm adopt Bagging mechanism. The sampling frequency is consistent with the original dimension. The number of sub-learners is set to 100. When MDMSE selecting sub-learners, the final number of selected sub-learners is 40. The results in table 1 are the average value of 30 independent experiments.

Table 1. Performance comparison

Method	AR	MR	Detection time/s
KELM	0.887	0.227	0.037
KELM integration	0.962	0.168	0.234
BL-KELM	0.985	0.114	0.128

It can be seen from table 1, the AR and MR of traditional KELM integration algorithm improves by 80% and decreases by 0.6%, respectively compared with KELM. But the detection time of traditional KELM integration increases by 10%. BL-KELM effectively reduces the number of integrated sub-learners, thus eliminating the impact of weak learners on the overall integrated learners, because it based on MDM criteria selects KELM sub-learners with good performance to integrate, which improves AR, reduces MR and detection time.

2) The influence of parameter setting on performance of BL-KELM.

The parameters that affect the performance of BL-KELM include the random extracted feature number F and the integrated number of sub-learners U when Bagging is used for random sampling. The effect of different F on the performance is shown in table 2.

Table 2. The effect of F on performance of BL-KELM

F	N	0.5N	0.01N	Log2(N)
AR	0.9855	0.9851	0.9842	0.9779
MR	0.124	0.121	0.117	0.115

As can be seen from table 2, F has little effect on the experimental results. The main reason is that the number of KDD99 sample sets used for training and testing is huge, and the algorithm has good generalization ability. It is not helpful to improve the algorithm only by adjusting F. However, F parameter adjustment can greatly improve the performance of learners trained by small sample sets. The number of sub-leaner determines the generalization performance of the integrated algorithm. But too many sub-learners will take up too many resources. In this experiment, the initial number of KELM sub-learners is 100, and the final integration number is determined by selective learning. The impact of the final number of selective integrator learners on intrusion detection performance is shown in table 3.

Table 3. The effect of U on performance of BL-KELM

U	20	40	60	80	100
AR	90.4%	98.2%	94.5%	94.1%	91.3%

It can be seen from table 3, with the increase of sub-learners, the accuracy of intrusion detection first gradually increases and then decreases. When the sub-classifier is 40, the network intrusion detection can achieve a high accuracy rate. However, with the further increase of the sub-classifier

number, the performance of intrusion detection has little impact. Therefore, in the subsequent comparison experiment, this paper adopts 40 as the final integration number of selective integrated learning.

3) Comparison experiments: The experiment is trained on the KDD99 data set by calling multiple single learner models in the Scikit-learn framework and common integrated algorithm. AR and MR are shown in table 4. Table 5 shows the time costs with different algorithms including WPFH [28], NB [29], FDS [30], and IDBN [31]. In this paper, BL-KELM adopts the optimal parameter setting in the previous section. In order to avoid errors caused by unstable single experiment, table 4 and table 5 show the average experimental results with 30 independent repeated experiments.

Table 4. Comparison results

Method	Evaluation index	Classification categories					
		PROBING	DOS	R2L	U2L	NORMAL	AVERAGE
WPFH	AR	0.924	0.931	0.948	0.933	0.932	0.932
	MR	0.675	0.615	0.611	0.627	0.612	0.615
NB	AR	0.932	0.927	0.928	0.932	0.961	0.943
	MR	0.636	0.578	0.582	0.617	0.559	0.606
FDS	AR	0.922	0.796	0.784	0.788	0.804	0.786
	MR	0.243	0.414	0.389	0.416	0.379	0.405
IDBN	AR	0.786	0.919	0.923	0.921	0.943	0.921
	MR	0.412	0.223	0.195	0.198	0.214	0.212
BL-KELM	AR	**0.982**	**0.986**	**0.991**	**0.992**	**0.992**	**0.986**
	MR	**0.113**	**0.133**	**0.109**	**0.107**	**0.107**	**0.122**

Table 5. Running time comparison

Method	Train time/s	Test time/s
WPFH	22.326	1.468
NB	13.986	5.227
FDS	11.784	4.765
IDBN	5.821	0.513
BL-KELM	**0.583**	**0.102**

The results in table 4 show that the BL-KELM method can effectively improve the accuracy of learner detection, and greatly reduce the MR, training and detection time. Table 5 shows that IDBN and IDBN have a good performance in the accuracy of intrusion recognition, but the training time is long. BL-KELM has a high correct detection rate, and in terms of time, the proposed algorithm spends less time.

The KDD99 test set contains 17 types of abnormal network connections not included in the training set. This experiment further verifies the generalization of BL-KELM by counting the recognition effects of the single learner algorithm and the common integrated learning algorithm on the unknown types of intrusive connections. The experimental results are shown in table 6.

Table 6. Unknown intrusion detection with different methods.

Method	Detected type number	MR of unknown type
WPFH	9	0.58
NB	13	0.29
FDS	14	0.27
IDBN	15	0.24
BL-KELM	**18**	**0.12**

It can be seen from table 6 that BL-KELM has good generalization performance and still maintains a low missing rate with unknown intrusion types.

3.2. Complex hybrid network physical simulation experiment

The performance of BL-KELM is further verified by building a network simulation platform to simulate network connection requests and abnormal intrusion requests in real life. In this simulation experiment, 30 physical terminal devices, a network server, seven routers and six switches are used to build a network physical simulation platform. The 30 physical terminal devices are divided into six sub-networks, each sub-network contains three Ethernet terminals, two wireless network terminals, one sub-network and a switch. Three Ethernet terminals are connected to the subnet route through the wired network switch, and then the switch is connected to the subnet route. Two wireless terminals are connected to the subnet route through Wi-Fi. The six subnets are connected to the attack server by a central router. On the attack server, Ettercap software is used to launch network simulation attacks on six sub-networks. The simulation attacks include PROBE, U2L, DOS and R2L. TCP dump is used for network data monitoring on 30 physical terminals, network connection data is obtained by capturing packets. Log information on terminal devices is collected at the same time. The collected information including 41 features forms a simulation data set. Finally, the processed data set is sent to the BL-KELM detector in real time for judgment that whether it needs to receive the network connection request.

Through a week of data collection, 30000 network requests are filtered and counted every day, 9000 requests are connected with attacks containing 1000 unknown attacks. The average detection results are shown in table 7.

Table 7. Simulation results

Type	AR	MR	Average response time/s
DOS	0.984	0.14	0.10
R2L	0.983	0.12	0.08
U2R	0.991	0.08	0.07
PROBING	0.987	0.11	0.09
Unknown intrusion	0.981	0.19	0.08

As can be seen from table 7, the BL-KELM method has better effect in network intrusion detection, especially for unknown network intrusion connections, its recognition rate exceeded 98%, and the average response time is controlled within 0.1s, which satisfies the effectiveness and real-time performance of network intrusion detection.

4. Conclusion

The proposed BL-KELM uses MDMSE to calculate integration gain of each KELM sub-learners. Partial selective integration is performed by selecting KELM sub-learners with high gain. BL-KELM not only has the efficient learning characteristics of KELM, but has the generalization ability of bagging integration algorithm. It shows good real-time performance in both training and testing, and can detect abnormal network connections in time. The experimental results show that BL-KELM can effectively detect various known and unknown intrusion types on both the public KDD99 data set and the manually built complex hybrid network physical simulation platform, it has a very low false alarm rate and missing alarm rate at the same time.

References

[1] Li P, Chen Z, Yang L T, et al. An Incremental Deep Convolutional Computation Model for Feature Learning on Industrial Big Data. IEEE Transactions on Industrial Informatics, vol.15, no.3, pp. 1341-1349, 2019.

[2] Laghari AA, Wu K, Laghari RA, Ali M, Khan AA. A Review and State of Art of Internet of Things (IoT). Archives of Computational Methods in Engineering. 2021 Jul 14:1-9.

[3] Behrisch M, Streeb D, Stoffel F, et al. Commercial Visual Analytics Systems-Advances in the Big Data Analytics Field. IEEE Transactions on Visualization & Computer Graphics, 2019, 25(10):3011-3031.

[4] Nazir R, Kumar K, David S, Ali M. Survey on Wireless Network Security. Archives of Computational Methods in Engineering. 2021 Jul 13:1-20.

[5] Rathore D, Jain A. Design Hybrid method for intrusion detection using Ensemble cluster classification and SOM network. International Journal of Advanced Computer Research, 2019, 2(3):181-186.

[6] Homoliak I, Martin Teknös, Martín Ochoa, et al. Improving Network Intrusion Detection Classifiers by Non-payload-Based Exploit-Independent Obfuscations: An Adversarial Approach. Security & Safety, 2019, 5(17):156245.

[7] Papamartzivanos, Dimitrios, Gomez, et al. Dendron: Genetic trees driven rule induction for network intrusion detection systems. Future Generations Computer Systems, 2018.

[8] Verma A, Ranga V. Statistical analysis of CIDDS-001 dataset for Network Intrusion Detection Systems using Distance-based Machine Learning. Procedia Computer Science, 2018, 125:709-716.

[9] Laghari AA, Jumani AK, Laghari RA. Review and State of Art of Fog Computing. Archives of Computational Methods in Engineering. 2021 Feb 1:1-3.

[10] Li X, Xiang S, Zhu P, et al. Establishing a Dynamic Self-Adaptation Learning Algorithm of the BP Neural Network and Its Applications. International Journal of Bifurcation & Chaos, 2015, 25(14):1540030.

[11] Jing Yu, Hang Li. Modified Immune Evolutionary Algorithm for IoT Big Data Clustering and Feature Extraction Under Cloud Computing Environment. Journal of Healthcare Engineering, 1, 2020.

[12] Lin Teng, Hang Li and Shahid Karim. DMCNN: A Deep Multiscale Convolutional Neural Network Model for Medical Image Segmentation. Journal of Healthcare Engineering, 2019.

[13] Xiaowei Wang, Shoulin Yin, Hang Li. A Network Intrusion Detection Method Based on Deep Multi-scale Convolutional Neural Network. International Journal of Wireless Information Networks. 27(4), 503-517, 2020.

[14] Yin, S., Li, H. GSAPSO-MQC: medical image encryption based on genetic simulated annealing particle swarm optimization and modified quantum chaos system. Evolutionary Intelligence (2020). doi: 10.1007/s12065-020-00440-6

[15] Lim J S, Lee S, Pang H S. Low complexity adaptive forgetting factor for online sequential extreme learning machine (OS-ELM) for application to nonstationary system estimations. Neural Computing & Applications, 2013, 22(3-4):569-576.

[16] Zhang P, She K. A Novel Hierarchical Clustering Approach Based on Universal Gravitation. Mathematical Problems in Engineering, 2020, 2020.

[17] XX Wang, Zhang Y F, Xie J, et al. A density-core-based clustering algorithm with local resultant force. Soft Computing, 2020, 24(8).

[18] Gupta S, Deep K. A hybrid self-adaptive sine cosine algorithm with opposition based learning. Expert systems with applications, 2019.

[19] You T, Lei D, Cai L, et al. Parameter Identification of Fractional Order Chaotic System via Opposition Based Learning Bare-Bones Imperialist Competition Algorithm. International Journal of Computational Intelligence Systems, 2020.

[20] Chen Wang, Lin Liu, Chengcheng Xu, Weitao Lv. Predicting Future Driving Risk of Crash-Involved Drivers Based on a Systematic Machine Learning Framework.. International Journal of Environmental Research & Public Health, 16(3):334.2019. doi: 10.3390/ijerph16030334.

[21] Zhen Z, Yibing L, Shanshan J, et al. Modulation Signal Recognition Based on Information Entropy and Ensemble Learning. Entropy, 2018, 20(3):198-.

[22] Shay Vargaftik, Isaac Keslassy, Ariel Orda, Yaniv Ben-Itzhak. RADE: Resource-Efficient Supervised Anomaly Detection Using Decision Tree-Based Ensemble Methods. arXiv:1909.11877

[23] Ashley A.Mikshowsky, Daniel Gianola, Kent A.Weigel. Assessing genomic prediction accuracy for Holstein sires using bootstrap aggregation sampling and leave-one-out cross validation. Journal of Dairy Science, Volume 100, Issue 1, January 2017, Pages 453-464.

[24] Kabir, Enamul, Jiankun Hu, Hua Wang, and Guangping Zhuo. "A novel statistical technique for intrusion detection systems." Future Generation Computer Systems 79 (2018): 303-318.

[25] Zhang, Fuyong, Yi Wang, Shigang Liu, and Hua Wang. "Decision-based evasion attacks on tree ensemble classifiers." World Wide Web 23, no. 5 (2020): 2957-2977.

[26] Zhang, Fuyong, Yi Wang, and Hua Wang. "Gradient Correlation: Are Ensemble Classifiers More Robust Against Evasion Attacks in Practical Settings?." In International Conference on Web Information Systems Engineering, pp. 96-110. Springer, Cham, 2018.

[27] Yin, Jiao, MingJian Tang, Jinli Cao, Hua Wang, Mingshan You, and Yongzheng Lin. "Vulnerability exploitation time prediction: an integrated framework for dynamic imbalanced learning." World Wide Web (2021): 1-23.

[28] Gang L. An optimization detection algorithm for complex intrusion interference signal in mobile wireless network. Discrete & Continuous Dynamical Systems, 2019, 12(4&5):1371-1384.

[29] Nuo Y. A novel selection method of network intrusion optimal route detection based on naive Bayesian. International journal of applied decision ences, 2018, 11(1):1-17.

[30] Thanuja R, Umamakeswari A. Unethical Network Attack Detection and Prevention using Fuzzy based Decision System in Mobile Ad-hoc Networks. Journal of Electrical Engineering & Technology, 2018, 13.

[31] Tian Q, Han D, Li K C, et al. An intrusion detection approach based on improved deep belief network. Applied Intelligence, 2020(3).