Intrusion Detection Method Based on Genetic Algorithm of Optimizing LightGBM

Zhanbo Li

College of Information Engineering
Zhengzhou University
Zhengzhou, Henan, China
iezbli@zzu.edu.cn

Xiaoyang Li †

College of Information Engineering
Zhengzhou University
Zhengzhou, Henan, China
†1211855254@qq.com

ABSTRACT

In response to the high-precision requirements of intrusion detection systems, an intrusion detection method based on genetic algorithm of optimizing LightGBM is proposed. The recursive feature elimination algorithm is used to select the optimal feature subset, and a weighted loss function is designed to solve the problem of imbalanced network traffic data. Aiming at the problem that the performance of LightGBM is greatly affected by parameters and the cumbersome parameter adjustment, the powerful global search capability of genetic algorithm is used to optimize LightGBM and automatically determine the optimal parameter combination. Using the CIC-IDS2017 data set experiment, the experimental results show that the accuracy of this method is as high as 99.88%, which has higher detection accuracy than other methods.

CCS CONCEPTS

• Computing methodologies→Machine learning algorithms;

• Security and privacy → Network security

KEYWORDS

Intrusion detection, LightGBM, Genetic algorithm, Parameter optimization, Feature selection, Data imbalance

1 Introduction

With the widespread application of network products in modern society, people are increasingly relying on the convenience brought by the rapid development of the network, but at the same time, many network fraud incidents and network attacks have also occurred. Maintaining network security has become particularly important. Intrusion detection system, as an active defense mechanism for network security, has been always the focus of the network industry. Existing intrusion detection systems can be divided into host-based intrusion detection systems (HIDS) and network-based intrusion detection systems (NIDS). HIDS detects intrusions by checking system information such as system log records, while NIDS analyzes the inflow and outflow of local network packets to detect intrusions.

GBDT is a boosting algorithm. The principle aims to combine multiple decision trees to obtain the final result. GBDT, XGBoost, and LightGBM have been applied to NIDS. The reference[1] uses MSMOTE for oversampling and GBDT as the classifier. The reference[2] uses Gini coefficient as feature selection, and particle swarm to optimize GBDT parameters. The reference[3] uses PCA combined with firefly algorithm to reduce the dimensionality features, and XGBoost as the classifier, the result is better than other machine learning algorithms. The reference[4] uses LightGBM's ability to efficiently process massive amounts of data to design a real-time intrusion detection system. The reference[5] proposes to use ADASYN to deal with imbalance, LightGBM as the intrusion detection system of the classifier, and conduct experiments on three data sets to obtain a higher accuracy rate. The above references can show that XGBoost and LightGBM, as powerful and fast machine learning methods, can achieve excellent results in intrusion detection problems. However, there are many LightGBM parameters, especially when faced with multi-classification tasks, the combination of parameters has a great impact on the classification performance of the model, and how to determine the best parameters is very important. Genetic algorithm is a heuristic search algorithm, which performs well in other classifier parameter optimization problems. The reference[6] uses genetic algorithm to optimize the punishment factor, kernel parameters, and the tube size of SVM, and the detection accuracy is better than random selection of parameters. The reference[7] uses genetic algorithms to optimize the DBN structure, reducing training time and improving detection accuracy. The reference[8] uses genetic algorithm to optimize the number of clusters in GMM to avoid improper initialization of the number of clusters and cluster centers. In this paper, genetic algorithm is selected to automatically search and optimize LightGBM parameters to optimize classification performance. Meanwhile, in view of the serious imbalance of network traffic, you can choose to solve it at the data level or the algorithm level. The references[5, 9] use over-sampling and under-sampling algorithms to solve the problem at the data level. The reference[10] uses a weighted extreme learning machine to assign weights for different classes on the extreme

EITCE 2021, October 22–24, 2021, Xiamen, China

ACM ISBN 978-1-4503-8432-2/21/10…$15.00
https://doi.org/10.1145/3501409.3501651

learning machine to improve the detection rate of attack classes. The reference[11] uses a combination of stratified sampling and weighted support vector machines to improve minority and overall accuracy. Considering that the sampling algorithm is easy to increase the computational burden, generate noisy data, and cause information loss, etc., the LightGBM weighted by the loss function class is selected to solve the imbalance problem at the algorithm level.

Section 2 introduces the method design and application algorithm of optimizing LightGBM based on genetic algorithm. Section 3 introduces the data set, evaluation indicators, and experimental results. Section 4 is the conclusion.

2 Method Design

The flow chart of the intrusion detection model proposed in this paper based on genetic algorithm of optimizing LightGBM is shown in Figure 1. The related algorithms are introduced as below.

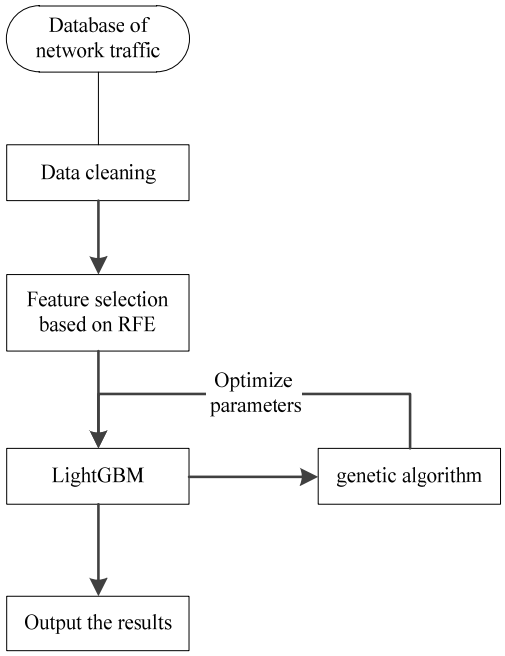

Figure 1: Method flow chart

2.1 LightGBM

LightGBM[12] is an improved gradient boosting decision tree (GBDT), which occupies less memory, with faster speed and higher accuracy than traditional GBDT. It can solve problems more quickly and efficiently when faced with massive data or high-dimensional data. The main improvements and advantages of LightGBM are as follows:

2.1.1 Histogram. The histogram algorithm handles feature boxing, that is, discretizes continuous feature values into bin data, and the time complexity of finding the best split point is greatly improved, and the memory usage is only 1/8 of the pre-sorting algorithm. The histogram algorithm can also speed up the difference, which can effectively construct the speed of the histogram.

2.1.2 EFB. On account of high-dimensional sparse data, mutually exclusive features are combined, which means that two or more features will not have non-zero values at the same time. Gaowei sparse data can be converted into low-latitude dense data, thereby reducing the amount of sample feature calculations.

2.1.3 Leaf-wise. The split strategy adopted by the traditional GBDT is level-wise, without paying special attention to the split gain of each leaf. When the same number of layers is generated, the calculation overhead is large, while LightGBM uses leaf-wise split strategy with depth limitation. Each layer only selects the leaf node with the largest split gain to split, which can make the loss drop with the fastest speed.

2.2 Recursive feature elimination

Recursive feature elimination (RFE) is a wrapped feature selection method. Each time RFE builds a model, the features in the current feature subset are sorted and the feature with the lowest score is eliminated. It is required to repeat the step of building a model to eliminate features until the number of features in the feature subset reaches the required number, and the resulting feature subset is the optimal feature subset. This paper chooses the feature importance of LightGBM as the scoring standard of RFE, and the calculation method of feature importance is: the sum of the information gain (Gain) generated by the feature split when the LightGBM model is constructed. Supposing that the feature X_d represents the d-th feature of the sample, then the split gain $Gain_{d,j}$ calculation method of X_d on the leaf node j is as shown in Formula (1):

$$Gain_{d,j} = \frac{1}{2}\left[\frac{(\sum_{i\in I_L} g_i)^2}{\sum_{i\in I_L} h_i + \lambda} + \frac{(\sum_{i\in I_R} g_i)^2}{\sum_{i\in I_R} h_i + \lambda} - \frac{(\sum_{i\in I_j} g_i)^2}{\sum_{i\in I_j} h_i + \lambda}\right] - \gamma \tag{1}$$

Where g_i is the first derivative of the loss function, h_i denotes the second derivative of the loss function, λ indicates the regularization coefficient, and γ refers to the minimum information gain required during split. I_j, I_L and I_R respectively represent the collection of samples that fall on the node and on the left and right of the node after split.

Let

$$G_j = \sum_{j\in I_j} g_j \tag{2}$$

Substitute Formula (2) into Formula (1) to get:

$$Gain_{d,j} = \frac{1}{2}\left[\frac{G_L^{\,2}}{H_L + \lambda} + \frac{G_R^{\,2}}{H_R + \lambda} - \frac{G_j^{\,2}}{H_j + \lambda}\right] - \gamma \tag{3}$$

The total split gain of the feature X_d is :

$$Gain_{d,J} = \sum_{j\in J} Gain_{d,j} \tag{4}$$

The j represents the collection of all leaf nodes. According to $Gain_{d,J}$, the feature importance is calculated, and features with low importance are recursively deleted.

2.3 Weighted loss function

LightGBM has the advantages of fast running speed and high classification accuracy, but in the terms of severely unbalanced data, especially in the case of multiple classifications, the classification performance needs to be further improved. When dealing with imbalanced classification problems, it is easier to learn the classes with a large number of samples during training, and ignore the classes with a small number of samples, which affects the overall performance of the classifier. Aiming at this problem, a weighted loss function is designed, and the class loss weight coefficient is introduced to improve LightGBM.

LightGBM is a kind of gradient boosting tree. The predicted label $\hat{y}_i$ of the sample i in the data set is predicted by the accumulation of multiple trees. The loss function of the t-th tree can be expressed as:

$$\begin{aligned} L^{(t)} &= \sum_{i=1}^{N} L(y_i, \hat{y}_i^{(t)}) \\ &= \sum_{i=1}^{N} [f_t(x_i) \frac{\partial L(y_i, \hat{y}_i^{(t-1)})}{\partial \hat{y}_i^{(t-1)}} + \frac{1}{2} f_t^2(x_i) \frac{\partial^2 L(y_i, \hat{y}_i^{(t-1)})}{\partial^2 \hat{y}_i^{(t-1)}}] \end{aligned} \tag{5}$$

The y_i is the true label of the sample, $\hat{y}_i$ is referred to the predicted label, and N stands for the total number of samples, $f_t(x_i)$ represents the prediction result of the t-th tree. When LightGBM performs multi-classification tasks, the weighted loss function is:

$$L_{new}^{(t)} = -\sum_{i=1}^{N} \sum_{m=0}^{M-1} \alpha_m y_{i,m} \ln(p_{i,m}^{(t)}) \tag{6}$$

Where, the prediction probability is:

$$p_{i,m}^{(t)} = \frac{e^{\hat{y}_{i,m}^{(t)}}}{\sum_{c=0}^{M-1} e^{\hat{y}_{i,c}^{(t)}}} \tag{7}$$

When the sample i is actually the k-class, $y_{i,k} = 1$ (8)

otherwise, $y_{i,k} = 0$ (9)

The α_m indicates the class loss weight coefficient. The first-order derivative and the second-order derivative can be obtained based on Formula (6) and (7):

$$g_{i,m_{new}} = \alpha_k (p_{i,m}^{(t-1)} - y_{i,m}) (m = k \Rightarrow y_{i,m} = 1) \tag{10}$$

$$h_{i,m_{new}} = \alpha_k p_{i,m}^{(t-1)} (1 - p_{i,m}^{(t-1)}) \tag{11}$$

Substitute $g_{i,m_{new}}$ and $\mathrm{h}_{i,m_{new}}$ into Formulas (1) and (3), the new spit gain formula is obtained as follows:

$$\begin{aligned} Gain_{new} = \frac{1}{2} [\frac{G_{L_{new}}^2}{H_{L_{new}} + \lambda} + \frac{G_{R_{new}}^2}{H_{R_{new}} + \lambda}] - \\ \frac{1}{2} \frac{(G_{L_{new}} + G_{R_{new}})^2}{H_{L_{new}} + H_{R_{new}} + \lambda} - \gamma \end{aligned} \tag{12}$$

It can be deduced from Formula (12) that the calculation of split gain is influenced by the first-order derivative and the second-order derivative during the node splitting. The first-order derivative and the second-order derivative of loss function are affected by the class loss weight coefficient, that is, by adding the loss weight of a minority of class samples, the minority of class samples will be more focused, and their detection rate is increased, thus optimizing the overall effects.

2.4 Genetic algorithm

Genetic algorithm is a heuristic search algorithm that draws on natural selection and genetic mechanisms in the evolution of organisms. It converts the solution to problems into the process of selection, crossover and mutation of chromosomal genes, sets fitness functions, and retains high fitness in the evolution process. Individuals are passed on to the next generation through cross-mutation operations to obtain the optimal solution. Genetic algorithm has the advantages of fast search speed, strong global optimization ability and high efficiency. Facing the situation where there are many LightGBM parameters, the model performance is greatly affected by the parameters, and the parameter adjustment is troublesome, genetic algorithm can show strong optimization ability, determine the optimal parameter combination, and optimize classification performance and detection accuracy. Figure 2 shows the optimization flow chart, and the detailed steps are described as below:

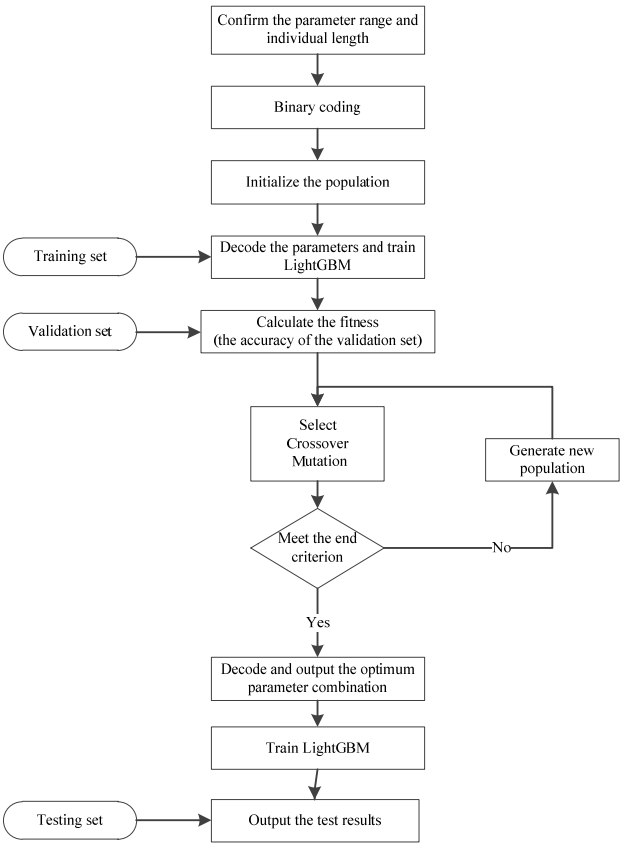

Figure 2: Flow chart of genetic algorithm of optimizing LightGBM

2.4.1 Parameters encoding. Convert the parameters of learning rate, n_estimators, max_depth, and num_leaves of LightGBM into binary codes, set the optimization range, and initialize the population.

2.4.2 Fitness evaluation. The fitness is defined as the accuracy rate, to encode the individuals in the population. The model is trained on the training set according to the decoded parameter values, and the prediction accuracy rate obtained on the validation is set as the fitness of the individual.

2.4.3 Genetic operation. Genetic operations include selection, crossover and mutation. The roulette algorithm is used to select individuals, and the probability of being selected is the proportion of each individual's fitness to the total fitness of all individuals in the population. Individuals with high fitness are easily selected. The selected individuals perform crossover and mutation operations according to the crossover rate and mutation rate to form a new population.

2.4.4 Output. Repeat the genetic operation until the number of iterations meets the end criterion, select the individual with the highest fitness, decode and output the best parameter combination, and substitute the best parameter combination for training and output the results of the test set.

3 Results and Analysis of Tests

The experiment is conducted on a computer of Windows 10 operating system, with physical memory of 62GB and the CPU model of Intel (R) Xeon (R) CPU E5-2678 v3 @ 2.50GHz, where the main installed software base is skrlearn 0.24.0。

3.1 Data sets and preprocessing

The CICIDS2017 data set[13] is a network traffic data set proposed by the Canadian Institute of Cyber Security (CIC). It contains 77-dimensional features and one column of labels. The data type includes one normal traffic and 14 attack traffic. This paper uses all data sets for experimentation. Table 2 describes the types of simulated attacks and their detailed statistics.

Table 1: Data description

Label	Class	Total
BENIGN	0	2273097
DDoS	1	128027
PortScan	2	158930
Bot	3	1966
Infiltration	4	36
Web Attack Brute Force	5	1507
Web Attack XSS	6	652
Web Attack Sql Injection	7	21
FTP-Patator	8	7938
SSH-Patator	9	5897
DoS slowloris	10	5796
DoS Slowhttptest	11	5499
DoS Hulk	12	231073
DoS GoldenEye	13	10293
Heartbleed	14	11

Preprocessing: Delete data from rows with infinite and missing values, and divide the data into training set, validation set, and test set by 2: 1: 1.

3.2 Evaluation indicators

In this paper, the accuracy ACC, accuracy PRE, detection rate DR, F1 and training time are taken as the evaluation indexes. Table 2 lists the confusion matrix.

Table 2: Confusion matrix

Predict value / True value	True	False
True(attack)	TN	FP
False(normal)	FN	TP

$$ACC = \frac{TP + TN}{ALL} \times 100\% \tag{13}$$

$$PRE = \frac{TP}{TP + FP} \times 100\% \tag{14}$$

$$DR = \frac{TP}{TP + FN} \times 100\% \tag{15}$$

$$F1 = \frac{2 \times PRE \times DR}{PRE + DR} \times 100\% \quad (16)$$

3.3 Results of tests

3.3.1 Results of tests based on RFE. In order to verify the RFE effect, eliminate redundant features, determine the number of feature selection, select the optimal feature subset, and avoid over-fitting on the training set, the accuracy is used as the evaluation index in this section to show the accuracy of the training set and the validation set under the different number of features in the RFE process. The first set of features with the highest accuracy obtained on the validation set is the selected optimal feature subset. The performance of the feature subset is verified on the test set. As is shown in Figure 3:

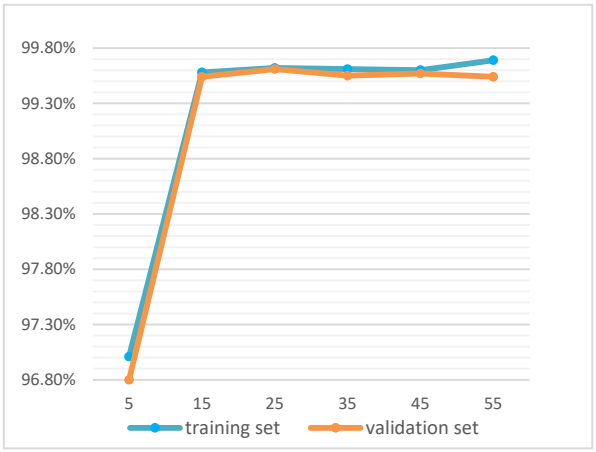

Figure 3 :Model accuracy under the change of RFE feature number

It can be concluded that when the number of features is 25, the accuracy of the validation rate is the highest, and then with the increase of the number of features, the accuracy of the training set increases, but the accuracy of the validation set decreases gradually, and the analysis may be overfitting on the training set. Therefore, 25 features are selected in this paper, and the feature subset of this group contains the following sequences:

[0,1,4,5,8,13,14,16,17,18,20,22,24,26,27,29,34,35,36,37,52,66 ,67,73,77]

The accuracies on the training set , validation set and test set are: 99.61%、99.61%、99.59%.

3.3.2 GA-LightGBM Test results.

3.3.2.1 Parameters setting. Parameters setting of genetic algorithm:Population size: 20; iteration times: 50; crossover probability: 0.6; mutation probability: 0.001; individual length:15.

LightGBM parameters searching optimization range and the optimum parameter are shown in Table 3.

Table 3: Optimum Parameter

Parameters	Optimum value	Searching optimization range
learning_rate	0.110	(0.01,0.165)
n_estimators	90	(20,95)
max_depth	11	(5,12)
num_leaves	100	(40,110)

3.3.2.2 Classification results. To show the classification performance of genetic algorithm of optimizing of LightGBM, Table 4 compares the overall accuracy, and the training and test time of the training set, validation set and test set before and after genetic algorithm optimization. Figures 4, 5 and 6 show PRE, DR and F1 of different categories before and after genetic algorithm optimization, and the detailed data after the genetic algorithm optimization.

Table 4 Accuracy and time before and after the optimization

	GA-LightGBM	LightGBM
Training-ACC	99.91%	99.61%
Validation-ACC	99.87%	99.61%
Testing-ACC	99.88%	99.59%
Training time	67s	84s
Testing time	11s	8s

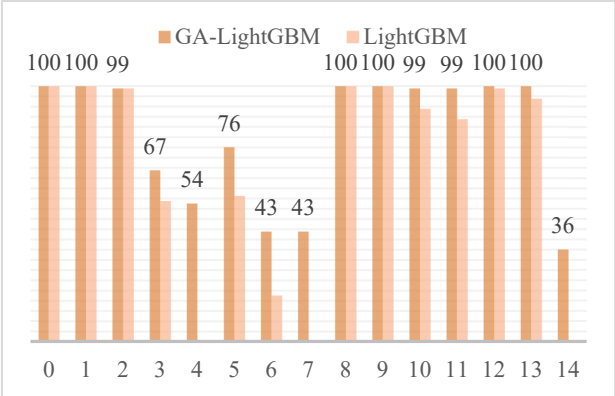

Figure 4: accuracy of different categories before and after the optimization

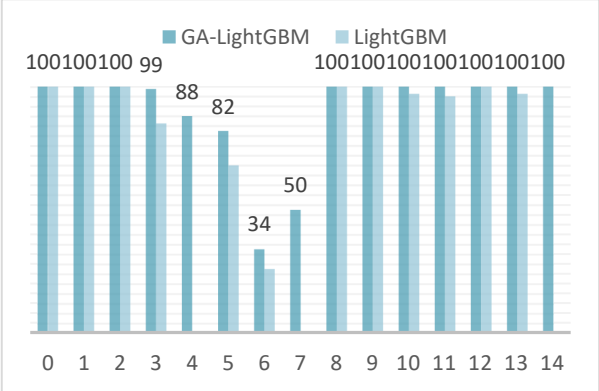

Figure 5: Detection rate before and after the optimization

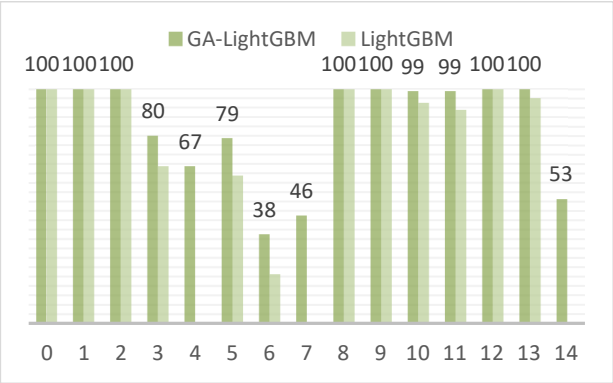

Figure 6: F1 of different categories before and after the optimization

According to the experimental results, the overall accuracy of the genetic algorithm before optimization reaches 99.59 %, but the recognition accuracy is very low on minority classes 4, 6, 7 and 14, and the PRE, DR and F1 are almost equal to 0, which proves that the classification accuracy of the optimized LightGBM needs to be further improved. On account of the classification results of genetic algorithm optimization, not only the overall accuracy rate reached 99.88 %, but also the PRE, DR and F1 of minority classes 4, 6, 7 and 14 were greatly improved, and the recognition accuracy of 3, 10, 11 and 13 classes was also improved to nearly 100 %. The optimized LightGBM training time is only 67 s, and the classification performance is good, especially DR, which can better detect the type of attack traffic.

3.4 Comparison with other methods

In order to further prove the effectiveness of this method, several advanced intrusion detection methods are compared with this method. Table 3 shows the accuracy comparison of the methods, and the proposed method is superior to the comparison of machine learning methods and deep neural network methods.

Table 5: Comparison of methods

Method	ACC (%)
GA-LightGBM	99.88
IGAN-CNN[14]	99.79
DNN[15]	99.61
DBN-KELM[16]	97.10
PCA-RF[17]	99.60

4 Conclusion

This paper proposes an intrusion detection method based on genetic algorithm of optimizing LightGBM. The recursive elimination algorithm is used to select 25-dimensional features and remove redundant features. The genetic algorithm is used to search the optimal parameter combination of four important parameters in LightGBM, which effectively improves the detection efficiency and accuracy. Experimental results on CICIDS2017 dataset show the feasibility and effectiveness of the proposed method.

REFERENCES

[1] J.-f. Cui, H. Xia, R. Zhang, B.-x. Hu, and X.-g. Cheng, "Optimization scheme for intrusion detection scheme GBDT in edge computing center," Computer Communications, vol. 168, pp. 136-145, 2021/02/15/, 2021.

[2] L. Li, Y. Yu, S. Bai, J. Cheng, and X. Chen, "Towards Effective Network Intrusion Detection: A Hybrid Model Integrating Gini Index and GBDT with PSO," Journal of Sensors, vol. 2018, pp. 1578314, 2018/03/26, 2018.

[3] S. Bhattacharya, S. R. K. S, P. K. R. Maddikunta, R. Kaluri, S. Singh, T. R. Gadekallu, M. Alazab, and U. Tariq, "A Novel PCA-Firefly Based XGBoost Classification Model for Intrusion Detection in Networks Using GPU," Electronics, vol. 9, no. 2, pp. 219, 2020.

[4] D. Jin, Y. Lu, J. Qin, Z. Cheng, and Z. Mao, "SwiftIDS: Real-time intrusion detection system based on LightGBM and parallel intrusion detection mechanism," Computers & Security, vol. 97, pp. 101984, 2020/10/01/, 2020.

[5] J. Liu, Y. Gao, and F. Hu, "A fast network intrusion detection system using adaptive synthetic oversampling and LightGBM," Computers & Security, vol. 106, pp. 102289, 2021/07/01/, 2021.

[6] F. Kuang, W. Xu, and S. Zhang, "A novel hybrid KPCA and SVM with GA model for intrusion detection," Applied Soft Computing, vol. 18, pp. 178-184, 2014/05/01/, 2014.

[7] Y. Zhang, P. Li, and X. Wang, "Intrusion Detection for IoT Based on Improved Genetic Algorithm and Deep Belief Network," IEEE Access, vol. 7, pp. 31711-31722, 2019.

[8] J. Liu, W. Zhang, Z. Tang, Y. Xie, T. Ma, J. Zhang, G. Zhang, and J. P. Niyoyita, "Adaptive intrusion detection via GA-GOGMM-based pattern learning with fuzzy rough set-based attribute selection," Expert Systems with Applications, vol. 139, pp. 112845, 2020/01/01/, 2020.

[9] T. Lu, Y. Huang, W. Zhao, and J. Zhang, "The Metering Automation System based Intrusion Detection Using Random Forest Classifier with SMOTE+ENN." pp. 370-374.

[10] J. Sharma, C. Giri, O.-C. Granmo, and M. Goodwin, "Multi-layer intrusion detection system with ExtraTrees feature selection, extreme learning machine ensemble, and softmax aggregation," EURASIP Journal on Information Security, vol. 2019, no. 1, pp. 15, 2019/10/22, 2019.

[11] M. Awad, and A. A, "Using weighted Support Vector Machine to address the imbalanced classes problem of Intrusion Detection System," KSII Transactions on Internet and Information Systems, vol. 12, 10/30, 2018.

[12] G. L. Ke, Q. Meng, T. Finley, T. F. Wang, W. Chen, W. D. Ma, Q. W. Ye, and T. Y. Liu, "LightGBM: A Highly Efficient Gradient Boosting Decision Tree," ADVANCES IN NEURAL INFORMATION PROCESSING SYSTEMS 30 (NIPS 2017), vol. 30, 2017.

[13] I. Sharafaldin, A. Habibi Lashkari, and A. Ghorbani, Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization, 2018.

[14] S. Huang, and K. Lei, "IGAN-IDS: An imbalanced generative adversarial network towards intrusion detection system in ad-hoc networks," Ad Hoc Networks, vol. 105, pp. 102177, 2020/08/01/, 2020.

[15] M. N, and J. B, "A deep learning based HTTP slow DoS classification approach using flow data," ICT Express, vol. 7, no. 2, pp. 210-214, 2021/06/01/, 2021.

[16] Z. Wang, Y. Zeng, Y. Liu, and D. Li, "Deep Belief Network Integrating Improved Kernel-Based Extreme Learning Machine for Network Intrusion Detection," IEEE Access, vol. 9, pp. 16062-16091, 2021.

[17] R. Abdulhammed, H. Musafer, A. Alessa, M. Faezipour, and A. Abuzneid, "Features Dimensionality Reduction Approaches for Machine Learning Based Network Intrusion Detection," Electronics, vol. 8, no. 3, pp. 322, 2019.