
Scalable Computing: Practice and Experience, ISSN 1895-1767, http://www.scpe.org
© 2025 SCPE. Volume 26, Issues 3, pp. 1284–1292, DOI 10.12694/scpe.v26i3.4207

# ALEX/ELM NETWORK DETECTION BASED ON IMPROVED FIREFLY SWARM OPTIMIZATION ALGORITHM

XIAOYAN WANG*

**Abstract.** To address the issues of blind spots and low detection accuracy associated with using a single machine learning approach in network intrusion detection, the author suggests employing an Alex/ELM network detection system enhanced by an optimized firefly swarm algorithm. In the construction of base classifiers, the differences between samples of each base classifier are increased by sampling the sample set and selecting the feature set; By employing various learning algorithms to boost the diversity of the base classifiers on the sample set, the detection results are combined through a weighting mechanism. An enhanced firefly optimization algorithm is used to fine-tune the classification result weights of each base classifier. Experimental results demonstrate that, compared to other algorithms, this approach maintains a relatively high detection accuracy (with a minimum accuracy of 95.5%), showcasing the algorithm's stability and effectiveness even with imbalanced samples. In conclusion, the method proposed by the author significantly enhances detection accuracy, while reducing both the false alarm rate and the missed alarm rate.

**Key words:** Intrusion detection, machine learning, heterogeneous integration, firefly optimization, Alex/ELM

**1. Introduction.** At present, at a time when a new round of scientific and technological industrial revolution and the risk of epidemic are intertwined and superimposed, the global network security situation is still grim. The frequent occurrence of network security threats against key industries, new technologies and new scenarios forces countries to continue to deepen security measures for key infrastructure and strengthen security risk prevention for new technologies and new applications. Cybersecurity legislation and law enforcement work together to fully defend cyberspace security [1]. Benefiting from policy intensification and the release of security needs, the development of network security technology has ushered in opportunities. Looking ahead to the 14th Five Year Plan period, the digital economy will shift towards a new stage of deepening development. In response to the new situation and challenges of network security, the connotation of network security concepts, new technologies, and security technology industries will usher in key developments [2]. The global cybersecurity situation remains strict, and the evolution and upgrading of network attack methods are not optimistic about the current global cybersecurity situation. On one hand, key industries like energy, transportation, and telecommunications have been increasingly targeted by cyber attacks. In 2021, numerous organizations were affected, including the largest oil pipeline operator in the United States, public railway operators in the UK, and major telecommunications companies in New Zealand. These attacks have caused multiple interruptions to network services and have had a profound impact on social stability and people's production and life [3]. On the other hand, there is an increasing number of network threats targeting new technologies and scenarios. Take the Internet of Vehicles as an example: the security risks associated with infrastructure components such as application platforms, network support, and data computing power are highly complex and multifaceted. According to the 2021 Global Automotive Network Security Report by Upstream, IoT infrastructure has emerged as a new target for cyber attacks. The percentage of connected vehicles subjected to cyber attacks rose to 77.6% from 2020 to 2021 alone. As network attack methods continue to evolve and improve, the conflict between network attack and defense has become increasingly intense [4]. Regarding attack methods, exploiting vulnerabilities to conduct chain attacks has become more common. In terms of tactics, enhanced network defense capabilities have made attacks more challenging. Consequently, attackers are now employing various strategies to bypass security measures and successfully infiltrate networks. When it comes to targets, driven by potential gains, attackers are increasingly selecting their targets with greater precision. By adopting intelligent methods, attackers begin to collect information about attack targets and target "high-value" targets to carry

*Luohe Medical College, Luohe, Henan, 462002, China (Corresponding author, XiaoyanWang7@163.com)

out attacks [5].

**2. Literature Review.** The process of network intrusion detection is actually the recognition of abnormal network behavior in the system. When there is a significant difference between the current behavior and normal behavior, an alarm message is issued. This process was early achieved through traditional machine learning algorithms. According to whether there are data labels involved in the training process, traditional machine learning algorithms can be divided into supervised learning and unsupervised learning. According to different operating mechanisms, supervised learning can be divided into generative methods and discriminative methods. The generation method starts from a statistical perspective and uses probability distribution to reflect the similarity between traffic data. Representative algorithms include naive Bayesian algorithm, Bayesian network, and hidden Markov model. Chang, W. Y. et al. introduced a hybrid metaheuristic algorithm that integrates dynamic multi-swarm particle swarm optimization with the firefly algorithm. This approach aims to achieve an optimal deployment solution that maximizes coverage and minimizes energy consumption using both static and mobile sensors. Moreover, the proposed algorithm incorporates a novel switching search mechanism between subgroups to prevent early convergence from becoming trapped in local optima. The simulation results show that compared with other PSO based deployment algorithms, this method can achieve better solutions in terms of coverage and energy consumption [6]. Gao, B. T. et al. developed a method for self-correcting the parameters of a disturbance rejection controller using an enhanced firefly swarm optimization algorithm. This algorithm incorporates local optimization operators based on sine and cosine functions, along with adaptive mutation strategies. The refined algorithm is then applied to tune the parameters of the disturbance rejection controller, enhancing the control system's anti-interference capabilities and ensuring parameter accuracy. The results indicate that optimizing disturbance rejection control with the improved firefly swarm optimization algorithm results in a quick response time, no overshoot, a stable tracking process, strong anti-interference capability, and superior optimization performance [7]. Zhou, X. et al. developed a multi-objective optimization model utilizing an enhanced firefly algorithm. This model uses the partial load rate of each chiller unit and the cooling rate of the freezer as optimization variables to determine the ISAC system's minimal energy consumption loss rate and operating cost. Experimental results demonstrate that, compared to strategies based on constant proportion, particle swarm optimization, and the standard firefly algorithm, the optimization strategy based on the improved firefly algorithm (IFA) achieves significantly greater energy savings and economic benefits [8].

The author employs the firefly optimization algorithm to optimize the decision output weights of each base classifier, identifying the optimal weighting scheme to enhance the detection model's accuracy. By using ensemble learning methods as the central detection algorithm, the author enhances the detection model's performance by refining the construction of the base classifiers and the method of result fusion.

**3. Research Methods.**

**3.1. Heterogeneous Integration Algorithm of GSO.** To enhance the effectiveness of ensemble learning, the author employs heterogeneous ensemble learning techniques, integrating multiple learning models and increasing the diversity of training samples from the training dataset. These methods improve the generalization capability of ensemble learning, thereby ensuring high detection accuracy.

**3.1.1. Heterogeneous basis classifier generation.** Ensemble learning involves employing a finite set of learning machines to tackle the same problem, where the final output for a given input example is determined by aggregating the outputs of these machines within the ensemble. A prerequisite for ensemble classifiers to outperform individual classifiers is their individual accuracy and diversity. Current machine learning methods with robust generalization capabilities often meet accuracy requirements without stringent parameter considerations. The crucial focus lies in enhancing the diversity among base classifiers. Ensemble learning can be categorized into two methods based on the similarities and differences in the classification algorithms of the base classifier: isomorphic and heterogeneous ensembles. Isomorphic ensembles utilize the same learning algorithm across all base classifiers, varying only in parameters and selected samples. On the other hand, heterogeneous integration employs different classifiers and learning algorithms, effectively ensuring diversity among base classifiers. Hence, heterogeneous integration is adopted for investigating intrusion detection, emphasizing the importance of diverse classification methods in enhancing detection accuracy [9].

The differences between base classifiers are not only affected by the self classification algorithm mentioned above, but also by the selection of the dataset when constructing the base classifier. Ensemble learning can be classified based on dataset selection methods into Pattern Level ensemble, which involves various resampling techniques, and Feature Level ensemble, which focuses on selecting different sample features. Pattern Level ensemble utilizes methods like repeated sampling or altering sample distributions to create diverse training sets for each base classifier, enhancing their variability. Commonly used Bagging and Boosting methods fall under Pattern Level integration. Feature Level integration, on the other hand, targets scenarios with numerous features by selecting subsets that capture distinct problem properties for each base classifier's training set. Given the high-dimensional sample space in network intrusion detection, employing Feature Level integration is practical. To enhance base classifier accuracy, a heterogeneous ensemble construction method that combines Pattern Level and Feature Level approaches is employed.

**3.1.2. Firefly Algorithm Weight Optimization.** After acquiring the outputs from multiple base classifiers, it's essential to fuse these results to derive the ultimate detection outcome. Since intrusion detection revolves around binary classification, the output results of each base classifier are assigned as +1 and -1, representing normal and detected intrusion, respectively. Assuming there are n base classifiers, the detection result of the i-th base classifier is $y_i$, and the fused weight is $x_i$, the final detection result y is as follows 3.1:

$$y = sgn(\sum_{i=1}^{n} x_i y_i) \tag{3.1}$$

To ensure the model achieves its highest detection accuracy by determining the optimal x, the objective function for optimization is expressed as follows 3.2:

$$f(x_i) = max(acc), 0 < x_i < 1 \tag{3.2}$$

Among them, *acc* represents the testing and detection accuracy of the entire integrated model under different weights $x_i$. In order to solve this problem, the author chose the Firefly Optimization Algorithm (GSO) to solve it [10]. GSO is a typical swarm intelligence optimization algorithm, originally proposed by Krishnanand in 2005. Its basic idea is to simulate the movement of individuals with low brightness towards individuals with high brightness in firefly swarm activities, in order to achieve optimization.

Compared to other swarm intelligence algorithms, it has the advantages of simplicity, fewer parameters, and easy implementation, making it suitable for determining the weights of decision layers.

The current position of the i-th firefly in GSO, which has a weight of $x_i(t)$, has a fluorescence value of $I_i(t)$ at that position, and t is the number of iterations. The iterative update process of GSO is determined by both fluorescence and position updates.

The formula for updating the fluorescence value is as follows 3.3:

$$I_i(t) = (1 - \rho) l_i(t - 1) + \gamma f(x_i(t)) \tag{3.3}$$

Among them, $f(x_i(t))$ objective function fitness value, $\rho$ is the volatilization factor of fluorescein, $\gamma$ is the fluorescence renewal rate. The position update formula is as follows 3.4:

$$x_i(t + 1) = x_i(t) + s(\frac{x_j(t) - x_i(t)}{||x_j(t) - x_i(t)}||) \tag{3.4}$$

Among them, s is the movement step size, $x_j(t) - x_i(t)$ is the distance between firefly j and i.

During each iteration, the dynamic decision domain radius is updated as follows 3.5:

$$r_d^i(t + 1) = min\{r_S, max\{0, r_d^i(t) + \beta(n_i - |N_i(t)|)\}\} \tag{3.5}$$

Among them, $r_S$ is the perception radius, $\beta$ is the update rate, $|N_i(t)|$ is the number of fireflies within the neighborhood range. Through iteration, the optimal weight with the highest brightness will be ultimately found [11].

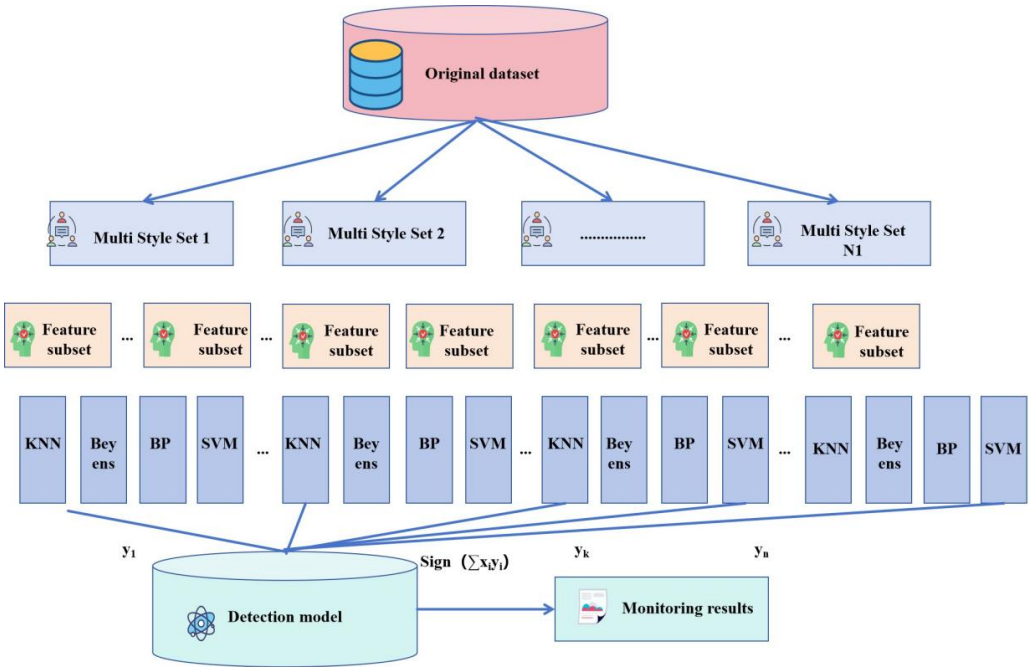


Fig. 3.1: Algorithm Process

**3.1.3. Algorithm process.** Building upon the previous analysis and considering the practicalities of intrusion detection problems, the author opts for heterogeneous integration. This involves selecting well-established machine learning methods with strong generalization performance: SVM, KNN, AlexNet, and ELM serve as learning techniques for base classifiers. To further improve the variability between base classifiers, an ensemble learning detection algorithm for GSO optimization weights combining Pattern-Level and Feature-Level selection training set methods. The detailed algorithmic process is depicted in Figure 3.1.

Here are the specific steps:

1. Employ the Bagging method to resample the samples with replacement, generating multiple subsets of samples.
2. Utilize the Feature Level method to randomly select features from the previously generated sample subsets, obtaining feature subsets.
3. Train different learning methods on the feature subsets obtained in Step 2. The parameters for each base classifier are determined using a simple trial and error method, resulting in the creation of each base classifier.
4. Synthesize the results from each base classifier with weighted fusion. The weights are optimized using the firefly algorithm as described in Section 1.2. The final detection results are then produced through a sign function [12].

**3.2. Algorithm validation.** In order to verify the effectiveness of the algorithm, we will first start with its effectiveness and examine its classification performance by applying the algorithm to the current universal dataset. Here, German, Ionosphere, Image, and Thyoid datasets are selected as the experimental datasets. The author's proposed algorithm is compared against traditional classification methods to evaluate performance differences. Additionally, a comparison is made between Bagging and Boosting ensemble algorithms to assess their respective effectiveness. For ensemble learning, it is not advisable to have too many or too few integrated base classifiers. Select 40 base classifiers and construct them based on this method, bagging, and boosting, respectively. The author samples the samples in the first layer and performs 5 resamples on each sample. Based on this, two random feature selections are performed on each subset, with a feature selection ratio of

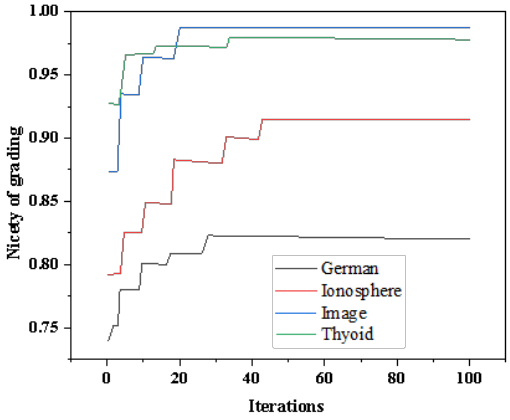


Fig. 3.2: GSO optimization results for different datasets

Table 3.1: Comparison of experimental accuracy for general datasets

| data set | Classification accuracy(Acc %) | | | | |
|---|---|---|---|---|---|
| | KNN | AlexNet | ELM | SVM | The method (Alex/ELM) |
| German | 74.5 | 76.7 | 75.8 | 77.5 | 81.4 |
| Ionosphere | 81.3 | 82.7 | 84.5 | 86.2 | 90.7 |
| Image | 92.5 | 94.3 | 95.2 | 95.8 | 98.2 |
| Thyoid | 93.1 | 94.7 | 94.2 | 95.0 | 97.1 |

80%. This resulted in 10 training sets. Based on these 10 training sets, SVM KNN, AlexNet, and BP neural networks were trained to obtain 40 base classifiers. The final classification result is obtained using the majority voting method [13,14].

Firstly, compare the classification performance of a single classifier and this method. For the single classifier, we employ base classifier methods, namely SVM, KNN, AlexNet, and BP neural networks. In SVM and BP neural networks, Gaussian functions are chosen as the kernel and activation functions, respectively. The parameters for each learner are determined through 5-fold cross-validation and grid search methods. The parameters for each base classifier in this algorithm are straightforwardly set without any special processing. Given the robust learning capabilities of the selected methods, the chosen parameters typically fulfill the requirements. In the GSO optimization algorithm, based on the problem, set the number of fireflies to 50, the fluorescence volatilization factor =0.4, the fluorescence update rate =0.6, and the update rate =0.08, neighborhood threshold =5, with 100 generations. The obtained experimental results are shown in Figure 3.2.

The graph depicts that the classification accuracy of each dataset remains consistently high during the initial stages, suggesting that the ensemble learning algorithm yields promising learning outcomes and maintains stability throughout the process. On different datasets, the algorithms converge quickly, indicating that the GSO algorithm has good convergence, and each algorithm ultimately stabilizes at a relatively high classification accuracy [15]. Next, the results of this article will be compared with several other single classification methods, and Table 1 presents the classification results.

The experimental findings reveal that among the algorithms tested, KNN exhibits the lowest classification accuracy. Despite its simplicity and high operational efficiency, the KNN algorithm tends to extract minimal classification information. In contrast, SVM, AlexNet, and ELM yield comparable classification results. These methods leverage distinct learning mechanisms, offering varied perspectives for sample learning and demonstrating proficiency in handling nonlinear classification tasks. The author's proposed method achieves optimal

Table 3.2: Comparison of Average Differences of Base Classifiers

| Data set | Average difference value | | |
|---|---|---|---|
| | Bagging | Boosting | This method (Alex/ELM) |
| German | 0.203 | 0.216 | 0.272 |
| Ionosphere | 0.170 | 0.181 | 0.2124 |
| Image | 0.013 | 0.012 | 0.020 |
| Thyoid | 0.064 | 0.073 | 0.103 |

Table 3.3: Classification accuracy results of integrated algorithms

| Data set | Classification accuracy(Acc%) | | | |
|---|---|---|---|---|
| | Bagging | Boosting | Voting fusion | GSO optimization |
| German | 78.4 | 78.4 | 80.3 | 81.3 |
| Ionosphere | 87.0 | 88.3 | 90.3 | 90.7 |
| Image | 97.3 | 97.1 | 97.7 | 98.1 |
| Thyoid | 95.3 | 95.8 | 96.4 | 97.1 |

classification accuracy by integrating diverse classifiers, thereby compensating for individual classifiers' errors and omissions. This ensemble learning strategy enhances the overall classification's generalization capability, showcasing the inherent advantage of ensemble learning.

Next, we'll assess the performance of this method against traditional ensemble learning algorithms. Traditional ensemble algorithms such as classic bagging and boosting integrate 40 base classifiers. The distinction among base classifiers serves as an indicator of ensemble learning effectiveness. The difference between base classifiers, denoted as D, is calculated using the following formula 3.6 on N samples:

$$Div = \frac{1}{TN}\sum_{t=1}^{T}\sum_{i=1}^{N} d_t(x_i) \tag{3.6}$$

Among them, the following equation 3.7:

$$d_t(x_i) = \begin{cases} 0 & if h_t(x_i) = f(x_i) \\ 1 & if h_t(x_i) \neq f(x_i) \end{cases} \tag{3.7}$$

Among them, $h_t(x_i)$ represents the predicted label of the t-th individual classifier on sample $x_i$, $f(x_i)$ is the predicted result after integrating all individual classifiers, using a difference threshold of at least 80% of the difference of the previous t individual SVMs, we evaluate the effectiveness of ensemble integration. A higher difference implies lower correlation between base classifiers, leading to improved integration outcomes [16]. We computed the average difference among three ensemble algorithms across various datasets, as summarized in Table 3.2.

Table 3.2 reveals that, on the whole, the disparity between Bagging and Boosting integration methods is not notably substantial. However, compared to the preceding methods, the average difference observed in this method is notably higher. This is because this method adopts heterogeneous integration. Below, we will examine the generalization performance of several ensemble algorithms by comparing their classification accuracy. Here, we'll delve into the GSO optimization employed by the author and the fusion decision results solely based on voting. Table 3.3 provides a comparison of the classification accuracy among these four methods.

From the experimental results, it can be seen that several ensemble algorithms have achieved certain improvements in classification accuracy compared to single learning methods. In the context of constructing the base classifier in this study, the classification accuracy of the voting fusion method surpasses that of traditional Bagging and Boosting. This improvement can be attributed to the increased diversity among base classifiers, leading to enhanced integration effects. Additionally, the GSO optimization method achieves the

Table 4.1: Results of Intrusion Detection Experiment

| method | Normal detection number | noise factor | Attack detection count | false negative | accuracy |
|---|---|---|---|---|---|
| SVM | 943 | 5.50 | 231 | 7.10 | 94.31 |
| Bagging | 973 | 2.50 | 236 | 5.10 | 97.31 |
| Boosting | 976 | 2.20 | 233 | 6.30 | 97.61 |
| The method (Alex/ELM) | 991 | 0.70 | 242 | 2.70 | 99.11 |

highest classification accuracy compared to simple voting methods [17,18]. These findings suggest that the method proposed in this article significantly enhances ensemble learning performance, thereby establishing a solid foundation for its application in intrusion detection.

**4. Result analysis.** The experiments conducted above validate the efficacy of the GSO-optimized ensemble learning algorithm proposed by the author in achieving high accuracy for binary classification tasks. This section extends the application of the algorithm to intrusion detection using the CSE-CIC-IDS2018 dataset. This dataset, collaboratively released by the Canadian Communications Security Establishment (CSE) and the Canadian Institute for Cybersecurity (CIC), comprises network traffic and system logs from 50 attack hosts targeting 420 computers and 30 servers across 5 departments within an enterprise. It encompasses a total of 7 types of attacks, including Brute force, Heartbleed, Botnet, DoS, DDoS, Web, and Infiltration. Compared to the traditional KDD99 dataset, CSE-CIC-IDS2018 can better simulate the network environment of enterprises, which includes a variety of network protocols and new attack methods. The original dataset is very large, with a ratio of approximately 4:1 between normal and invasive samples. In this experiment, training samples were selected in a 4:1 ratio. Specifically, 4000 normal samples and 1000 intrusion samples (including all 7 selected attack types) were randomly chosen from the training set. The test sample consists of 1000 normal samples and 250 attack samples. Individual machine learning methods, with SVM selected as the model, as well as ensemble learning algorithms such as bagging and boosting, are employed for detection experiments. It's worth noting that the base classifiers integrated with bagging and boosting algorithms are all SVM models. The performance metrics evaluated include the false detection rate, false alarm rate, and overall detection accuracy of each algorithm, as illustrated in Table 4.1.

The detection results presented in Table 4.1 highlight the comparatively high false detection and false alarm rates when solely employing the SVM algorithm for intrusion detection, underscoring the benefits of ensemble learning approaches. Among the three ensemble algorithms examined, the author's proposed algorithm demonstrates the most robust detection performance, aligning with our findings from experiments conducted on the UCI dataset. By enhancing the disparity between base classifiers, the algorithm's generalization performance is notably improved. Specifically, by analyzing the detection performance of different attack methods, we can observe that the algorithm proposed by the author is effective for Botnet The detection rate of attack types such as DoS, DDoS, and Web has basically reached 100%, and the main erroneous judgments are concentrated in the Brute force and Infiltration attack types. Analyzing the data of these two attacks, it was found that there are some normal traffic data in the abnormal samples, which have the same numerical values as the abnormal traffic in terms of characteristics. Comparing the performance of several methods on these two easily misclassified datasets, the method proposed by the author can greatly improve the detection accuracy of detection algorithms for these two types of attacks [19].

Moving forward, we'll utilize the Receiver Operating Characteristic (ROC) curve to evaluate the performance of the detection algorithms. When assessing a model's quality using the ROC curve, two key aspects are considered: the curve's shape and the Area Under Curve (AUC). A curve that approaches the upper left corner indicates superior detection performance, while a deviation from this corner suggests poorer performance. AUC represents the area beneath the ROC curve, serving as a reflection of the detection model's diagnostic value. A higher AUC value, closer to 1, signifies better model performance. Overall, the ROC curves of the algorithms employed earlier tend to cluster towards the upper left corner, indicative of their efficacy as detectors. Notably,

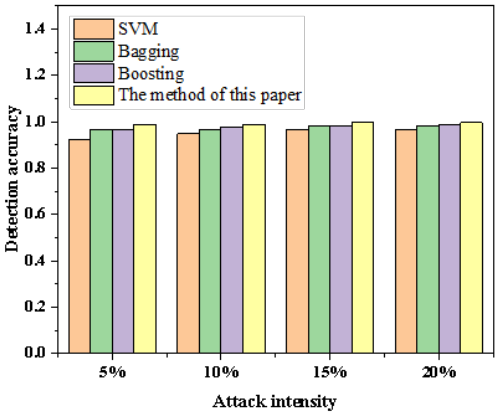


Fig. 4.1: Detection accuracy under different attack intensities

the author's proposed algorithm exhibits the most favorable bias, significantly outperforming other detection methods. Furthermore, in terms of AUC, this method boasts a larger area under the curve, reaching 0.992, surpassing the performance of the other three methods. In summary, the ROC curve analysis underscores the advantages of the method proposed by the author.

In real-world scenarios, network attacks often occur intermittently, with fluctuations in attack frequency over time. This variability is reflected in the dataset as the ratio of normal samples to attack samples. Consequently, collecting training datasets poses challenges, as a larger dataset theoretically leads to better detection performance. However, during the initial stages of detection, when a relatively small number of attack events are collected, the effectiveness of the detection model may be compromised. To assess the efficacy of detection models under different attack intensities, simulated scenarios are created with attack samples ranging from 5% to 20% of the total samples. The detection models are trained using these varying attack samples, and their detection accuracy under different learning modes is depicted in Figure 4.1.

Figure 4.1 illustrates that the trained detection models successfully identify intrusion events across varying attack intensities. When the attack intensity is low (i.e., a small proportion of attack samples in the training set, such as 5%), the SVM model achieves a detection accuracy of 91.4%. However, it exhibits a relatively high missed detection rate, primarily attributed to sample imbalance, causing the SVM's classification plane to deviate. From the experimental results, it can also be seen that sample imbalance can affect the effectiveness of machine learning. Compared with other algorithms, this method can consistently maintain a relatively high level of detection accuracy (with a minimum of 95.5%), indicating that the algorithm is stable and can also achieve good detection results in cases of imbalanced samples [20].

**5. Conclusion.** Given that network intrusion detection is essentially a binary classification problem with individual machine learning models often yielding suboptimal accuracy, a heterogeneous ensemble learning approach is adopted. By enhancing the disparity between the training sets and methods in the base classifiers, the overall integration effect is improved. Furthermore, to further enhance detection accuracy, enhancements have been made to the result fusion aspect of ensemble learning. The GSO algorithm is optimized to determine the optimal weight of the base classifier, and the final detection result is obtained through weighted fusion. Experimental results demonstrate the stability and accuracy of the proposed method, showcasing its practical value in real-world network intrusion detection applications. Addressing challenges associated with uneven training samples remains a focal point for future research efforts.

**6. Acknowledgements.**

1. The study was supported by Key scientific research projects of colleges and universities in Henan

Province“Research on the cultivation of College Teachers' information teaching ability from the perspective of TPACK”(Grant No.22A880031);
2. The study was supported by Academic Degrees & Graduate Education Reform Project of Henan Province”Research on the path of graduate students' Academic Integrity Construction”(Grant No. 2021SJGLX058Y).

REFERENCES

[1] Devaraj, A. F. S., Elhoseny, M., Dhanasekaran, S., Lydia, E. L., & Shankar, K. (2020). Hybridization of firefly and improved multi-objective particle swarm optimization algorithm for energy efficient load balancing in cloud computing environments. Journal of Parallel and Distributed Computing, 142, 36-45.
[2] Pitchaimanickam, B., & Murugaboopathi, G. (2020). A hybrid firefly algorithm with particle swarm optimization for energy efficient optimal cluster head selection in wireless sensor networks. Neural Computing and Applications, 32, 7709-7723.
[3] Igiri, C. P., Singh, Y., & Poonia, R. C. (2020). A review study of modified swarm intelligence: particle swarm optimization, firefly, bat and gray wolf optimizer algorithms. Recent Advances in Computer Science and Communications (Formerly: Recent Patents on Computer Science), 13(1), 5-12.
[4] Rahkar Farshi, T., & K. Ardabili, A. (2021). A hybrid firefly and particle swarm optimization algorithm applied to multilevel image thresholding. Multimedia Systems, 27(1), 125-142.
[5] Kaya, S., Gümüşçü, A., Aydilek, I. B., Karacizmeli, I. H., & Tenekeci, M. E. (2021). Solution for flow shop scheduling problems using chaotic hybrid firefly and particle swarm optimization algorithm with improved local search. Soft Computing, 25(10), 7143-7154.
[6] Chang, W. Y., Soma, P., Chen, H., Chang, H., & Tsai, C. W. (2023). A hybrid firefly with dynamic multi-swarm particle swarm optimization for wsn deployment. Journal of Internet Technology(4), 24-31.
[7] Gao, B. T., Shen, W., Dai, Y., & Ye, Y. (2022). Parameter tuning of auto disturbance rejection controller based on improved glowworm swarm optimization algorithm. Assembly Automation, 34(8), 6432-6440.
[8] Zhou, X., Yu, J., Zhang, W., Zhao, A., & Zhou, M. (2022). A multi-objective optimization operation strategy for ice-storage air-conditioning system based on improved firefly algorithm:. Building Services Engineering Research & Technology, 43(2), 161-178.
[9] Ab Talib, M. H., Mat Darus, I. Z., Mohd Samin, P., Mohd Yatim, H., Ardani, M. I., Shaharuddin, N. M. R., & Hadi, M. S. (2021). Vibration control of semi-active suspension system using PID controller with advanced firefly algorithm and particle swarm optimization. Journal of ambient intelligence and humanized computing, 12, 1119-1137.
[10] Kaya, S. (2023). A hybrid firefly and particle swarm optimization algorithm with local search for the problem of municipal solid waste collection: a real-life example. Neural Computing and Applications, 35(9), 7107-7124.
[11] Chou, D., & Jiang, M. (2021). A survey on data-driven network intrusion detection. ACM Computing Surveys (CSUR), 54(9), 1-36.
[12] Drewek-Ossowicka, A., Pietrołaj, M., & Rumiński, J. (2021). A survey of neural networks usage for intrusion detection systems. Journal of Ambient Intelligence and Humanized Computing, 12(1), 497-514.
[13] Ashiku, L., & Dagli, C. (2021). Network intrusion detection system using deep learning. Procedia Computer Science, 185, 239-247.
[14] Asif, M., Abbas, S., Khan, M. A., Fatima, A., Khan, M. A., & Lee, S. W. (2022). MapReduce based intelligent model for intrusion detection using machine learning technique. Journal of King Saud University-Computer and Information Sciences, 34(10), 9723-9731.
[15] Smys, S., Basar, A., & Wang, H. (2020). Hybrid intrusion detection system for internet of things (IoT). Journal of ISMAC, 2(04), 190-199.
[16] Mishra, N., & Pandya, S. (2021). Internet of things applications, security challenges, attacks, intrusion detection, and future visions: A systematic review. IEEE Access, 9, 59353-59377.
[17] Thakkar, A., & Lohiya, R. (2023). Fusion of statistical importance for feature selection in Deep Neural Network-based Intrusion Detection System. Information Fusion, 90, 353-363.
[18] Gümüşbaş, D., Yıldırım, T., Genovese, A., & Scotti, F. (2020). A comprehensive survey of databases and deep learning methods for cybersecurity and intrusion detection systems. IEEE Systems Journal, 15(2), 1717-1731.
[19] He, K., Kim, D. D., & Asghar, M. R. (2023). Adversarial machine learning for network intrusion detection systems: A comprehensive survey. IEEE Communications Surveys & Tutorials, 25(1), 538-566.
[20] Kumar, V., & Kumar, D. (2021). A systematic review on firefly algorithm: past, present, and future. Archives of Computational Methods in Engineering, 28, 3269-3291.

*Edited by:* Bradha Madhavan
*Special issue on:* High-performance Computing Algorithms for Material Sciences
*Received:* Jun 28, 2024
*Accepted:* Aug 6, 2024