IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 1 March 2023, accepted 17 March 2023, date of publication 21 March 2023, date of current version 24 March 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3259981

RESEARCH ARTICLE

Hybrid Firefly and Black Hole Algorithm Designed for XGBoost Tuning Problem: An Application for Intrusion Detection

XIN YONG[1] AND YUELIN GAO[2]

[1]School of Computer Science and Engineering, North Minzu University, Yinchuan, Ningxia 750021, China
[2]Ningxia Province Key Laboratory of Intelligent Information and Data Processing, North Minzu University, Yinchuan, Ningxia 750021, China

Corresponding author: Yuelin Gao (gaoyuelin@263.net)

This work was supported in part by the Key Project of Ningxia Natural Science Foundation under Grant 2022AAC02043, in part by the National Natural Science Foundation of China under Grant 11961001 and Grant 61561001, in part by the Construction Project of First-class Subjects in Ningxia Higher Education under Grant NXYLXK2017B09, in part by the Major Proprietary Funded Project of North Minzu University under Grant ZDZX201901, and in part by the Basic Discipline Research Projects supported by Nanjing Securities under Grant N-JZQJCXK202201.

ABSTRACT Computer networks have touched every aspect of human life, it cannot be overstated that cyber security is of great importance and significance. Intrusion detection techniques play an important role in the field of network security, but it also faces significant challenges. In this paper, we propose a Hybrid Firefly and Black Hole Algorithm (HFBHA) for parameter tuning of the XGBoost model and apply it to the study of intrusion detection systems. Firstly, the algorithm designs a double black hole mechanism by introducing the concept of the second black hole and adjusting the moving trajectory of the stars using the attraction of both black holes. Secondly, an improved initialization method of the stars is proposed, where a star that crosses the event horizon of the black hole has an opportunity to be replaced by a new star around the black hole. Finally, a combination of the firefly perturbation strategy and mutation operator is proposed to improve the global search capability of the algorithm. Both the effectiveness of the proposed method on the XBGoost parameter tuning problem and the feasibility of this strategy on intrusion detection applications are verified by comparison experiments based on the NSL-KDD dataset.

INDEX TERMS Black hole algorithm, firefly algorithm, intrusion detection, XGBoost.

I. INTRODUCTION

Nowadays, with the development of computer technology and the improvement of network infrastructure, computer network is widely used in all fields of human life [1]. However, the problem that cannot be ignored is malicious behaviors in the network environment which become threats to the availability, integrity, and confidentiality of computer systems. Internet and data security have aroused widespread concern in both industry and academia around the world [2]. It is an important task to establish a perfect computer network system, but the continuous expansion of computer networks and the complex technological environment due to malicious behaviors also challenge the goal [3]. Although data encryption, user authentication system, and other techniques have been extensively studied for solutions, malicious intrusion behaviors that threaten network security still occur frequently. These malicious attacks on the Internet result in serious economic losses for enterprises and individuals involved [4]. That is to say, network security problems still lack feasible and efficient solutions.

Network security techniques consist of anti-virus software, firewalls, and intrusion detection systems (IDSs) [5]. IDSs play a key role in protecting cyber security by monitoring and observing the behaviors of the computer to determine whether there exists an intrusion, and then warn the system administrators as well as provide possible solutions. There are three main categories of IDSs: misuse-based, anomaly-based,

The associate editor coordinating the review of this manuscript and approving it for publication was Sotirios Goudos.

and hybrid [6]. Misuse-based techniques work by detecting known attack patterns, but they are often ineffective in detecting unknown attacks. Anomaly-based techniques attempt to distinguish the attacks from normal activities through the deviation analyzed by algorithm models, which is possible to identify any attack theoretically. Hybrid techniques combine the advantages of both the former two methods. In addition to the above classification, IDSs can also be classified into host-based IDSs and network-based IDSs according to different monitoring agents [3].

Machine learning technology has been widely applied and developed in many fields such as industry, health care, education, and financial modeling in recent years [7]. Previous studies on anomaly detection systems are mostly based on machine learning algorithms [8]. For the aim of distinguishing the monitored behaviors into normal and abnormal behaviors, single or composite machine learning technology is trained as a classifier to analyze and diagnose. With improved machine learning algorithms, the IDSs can even detect new attacks that have not been recorded before in the database, such as zero-day threats.

Ensemble learning algorithms are a group of efficient machine learning methods, which improves the generalization ability of the model by combining the classification results of multiple base learners. The main popular ensemble techniques are boosting and bagging [2]. Recently, XGBoost, one of the gradient boosting algorithms that focus more on both speed and accuracy, has attracted much attention and has been widely used in many applications of machine learning [9], [10], [11]. The high efficiency and excellent performance of the XGBoost model in classification and regression tasks is the reason why it is selected for research in this paper. Although the XGBoost algorithm shows better performance than other algorithms, there are more parameters in this model. On the one hand, too many parameters may have the problem of not exploiting the optimal performance of the algorithm. On the other hand, it also increases the complexity of the work if the manual adjustment is used based on expert experience.

Swarm intelligence (SI) is an important class of meta heuristic algorithms that excel in solving global optimization problems. Related studies have shown its efficiency on parameter tuning problems of the XGBoost algorithm model [12], [13]. Therefore, in this paper, the black hole algorithm (BHA) and firefly algorithm (FA) are studied and a Hybrid Firefly and Black Hole Algorithm (HFBHA) is proposed. Then the proposed algorithm is utilized for the XGBoost parameter tuning problem so that it can be applied to building an IDS. In summary, the contributions of this paper are as follows:

(1) A double black hole mechanism is proposed, which assumes the impact of two black holes in the population on other stars. The concept of the second black hole is introduced and the moving formula of the stars is modified accordingly. The double black hole mechanism helps the algorithm avoid being manipulated into local optima due to the single influence of the black hole.
(2) An improved initialization method of the stars is designed, which makes full use of the currently known information of the optimal solution region. This strategy tends to search according to the priori information and extends the range of solution space that the algorithm can explore.
(3) Firefly perturbation strategy and mutation operator are introduced into the proposed method. The improved BHA, which is mixed with the FA, provides more selections for the way of star motion. The mutation operator for the optimal solution increases the probability of finding the global optimal solution.
(4) Experiments based on the train set and test set of NSL-KDD verify the effectiveness of the algorithm proposed in this paper. The advantages of the proposed method are proved by analyzing and discussing the results compared with other SI algorithms and machine learning methods.

The rest of this paper is organized as follows: section II shows the related work of SI in the field of IDs. Section III provides preliminary knowledge of BHA, FA, and XGBoost algorithms. The algorithm proposed in this paper is described in much more detail in section IV, while section V introduced the experimental results and analysis. Finally, the conclusion and the future research direction of this work are presented in section VI.

II. RELATED WORKS

It is common for heuristic algorithms to be applied to the optimization problems of various aspects of intrusion detection models. Authors in [14] made a detailed investigation on this topic and proposed a classification based on the applicability of these methods in improving the intrusion detection process. The article classifies these methods into feature selection/ reduction, weight selection/optimization, classification, and multi-objective intrusion detection model. This paper refers to the classification method given in [14]. However, for the sake of brevity, only research work more relevant to the content of this article will be presented in this section.

Feature selection can help the machine learning models achieve better performance, thus improving the classification accuracy of the obtained results. Alazzam et al. [15] proposed a wrapper feature selection algorithm using the pigeon inspired optimizer (PIO) for IDSs and evaluated the proposed algorithm using three popular datasets: KDD CUP 99, NLS-KDD, and UNSW-NB15. In [16], a wrapper feature selection strategy based on C4.5 and Bayesian network as a classifier using mutual information firefly algorithm (MIFA) was proposed and experimental results showed that a certain accuracy can be obtained with fewer features.

It is prevalent in machine learning areas that the weights and parameter values that need to be set carefully usually have

a significant impact on the performance of the algorithm. Therefore, the integration of the efficient optimization ability of SI can effectively improve the performance of the model. One of the widely studied models in this field is the support vector model (SVM). Enache and Patriciu [17] used the particle swarm optimization (PSO) algorithm and artificial bee colony (ABC) algorithm for optimizing the parameters C and ρ of the SVM model and applied it to the NSL-KDD dataset to improve the detection rate of the model and reduce the false alarm rate. Subsequently, Enache and Sgârciu [18] introduced the bat algorithm (BA) into the research field, effectively improving the parameter selection process of SVM. Kuang et al. [19] proposed an improved chaotic PSO to optimize the penalty factor C, kernel parameter σ, and tube size ε of the SVM model. The experimental results showed that the improved SVM model has lower computation time and higher prediction accuracy. Li et al. [20] proposed a velocity adaptive shuffled frog leaping bat algorithm (VASFLBA) to solve the problem that BA is prone to fall into local optima and lacks deep local search capability, and successfully applied it to the parameter tuning problem of SVM in industrial intrusion detection.

For other models with parameter optimization problems, SI can also play a superior role. Ali et al. [21] proposed a hybrid PSO to optimize the extreme learning machine (ELM) model, which outperforms the basic ELM. Yu et al. [22] discussed an optimized K-Means clustering algorithm based on the artificial fish swarm to overcome the sensitivity of the K-Means algorithm to the selection of initial clustering centers and obtain the optimal global clustering partition. Wei et al. [23] designed a joint optimization algorithm to optimize the network structure of the deep brief network (DBN). The experimental results show that it is an effective DBN-IDS approach. Qureshi et al. [24] proposed the ABC to train a random neural network (RNN) based system (RNN-ABC), trained it on NSL-KDD Train+, and tested it against undiscovered data.

In addition to the machine learning models mentioned above, the research on SI applied to the parameter tuning problem of the XGBoost model is also noteworthy. Wang et al. [25] proposed a method to optimize the XGBoost model with quantum-behaved particle swarm optimization (QPSO) and verified its effectiveness in network intrusion detection. Jiang et al. [26] proposed a PSO-XBGoost algorithm that constructs the optimal structure of the XGBoost model by PSO adaptively searching. The overall classification accuracy of their proposed model is higher than other alternative models and especially performs better in identifying minority groups of attacks. Song et al. [27] used the whale optimization algorithm (WOA) to find the optimal parameters of the XGBoost model and then applied the results to the KDD CUP 99 dataset. Zivkovic et al. [28] utilized a widely adopted modified version of FA applied to tune and optimize the XGBoost classifier hyperparameters and validated its effectiveness by the experiments on the widely used benchmark network intrusion detection datasets named NSL-KDD and the USNW-NB15. Based on the analysis of the above research work on XGBoost model parameter adjustment, it can be seen that the existing researches also have room for improvement. In the studies [25], [26], [27], [28], most of them still utilized the evaluation criteria that are widely used but may not sufficient such as precision and recall. For research of IDSs, the importance of missed detection rate and false alarm rate is much higher than those commonly used in machine learning model evaluation [29]. In addition, as the ''No Free Lunch'' theorem [30] reveals, no heuristic algorithm can achieve a superior performance than all the other algorithms. Therefore, although the excellent performance of SI algorithms has been studied and confirmed in relevant work, there is still a lack of effective algorithm research in this area. Therefore, this paper designs an improved SI algorithm to solve the problem of tuning the parameters of the XGBoost algorithm for providing a feasible solution for IDSs.

III. BACKGROUND METHODS

A. BLACK HOLE ALGORITHM

The black hole algorithm, proposed by Abdolreza [31] in 2013, is a SI algorithm inspired by black hole theory. According to the theory, there exist a certain number of stars in the universe space, in which the massive stars collapse to form a black hole. A black hole has so much matter concentrated that its gravitational field is so large that nothing around it can escape its gravity, even light [32], [33]. A mathematically defined surface around a black hole is called the event horizon, which is the limit that matter can reach.

Similar to other SI algorithms, the BHA requires setting an initial population of agents called ''stars''. In each iteration of the BHA, the best solution in the population is chosen as the black hole, which then starts to attract and make an effect on the trajectories of the other stars around it. Here, takes the minimum problem as an example, after defining the initial population and calculating the fitness values of all stars, the star with the smallest fitness value will be designated as the black hole. In this case, the updated formula for the other stars is shown in Eq. (1).

$$X_i^{t+1} = X_i^t + r \times \left(BH^t - X_i^t\right) \tag{1}$$

where X_i^t represents the location of the ith star in the tth iteration and X_i^{t+1} is its location in the $(t+1)$th iteration. BH^t is the location of the black hole in the tth iteration. r is a random number in the interval [0,1]. When the stars in the population are attracted by the black hole and move towards it, the distance of movement is determined by the random number r. During the movement of the stars, they may reach a better solution than the current black hole. In this case, the black hole is replaced by the better solution. Then the next star will begin to move towards this new black hole position. In addition, stars may also be swallowed by the black hole because they get too close to it (that is, they have crossed the event horizon radius). If it happens, a new star

will be randomly generated and placed in the search space to maintain the stability of the population. The calculation formula of the event horizon radius is shown in Eq. (2):

$$R = \frac{f_{BH}}{\sum_i^N f_i} \tag{2}$$

where f_i and f_{BH} represents the fitness values of the ith star and black hole respectively. N is the population size. In a certain iteration, for each star in the population, the distance from it to the black hole needs to be calculated during its movement to judge whether it has crossed the event horizon radius. When all the stars have been moved, a new round of iterations begins. The algorithm will continue to iterate until the convergence condition is met and the optimal solution is found. The pseudo-code of the black hole algorithm is shown in **Algorithm 1**.

Algorithm 1 Pseudo-Code of BHA

Input: The maximum number of iterations T, the population size N
Output: The optimal solution that the algorithm found
Initialize a population of stars.
Evaluate the swarm by the fitness function and set the best solution as the black hole.
For $t \leftarrow 1$ to T do
For $i \leftarrow 1$ to $(N-1)$ do
Update the star X_i using Eq. (1).
Evaluate the fitness value of the star $f(X_i)$.
If $f(X_i) < f(BH)$ then

$$BH = X_i$$

Calculate the event horizon R using Eq. (2).
If $\sqrt{(X_i - BH)^2} < R$ then
Replace the current star X_i with a new star in a random location in the search space.

B. FIREFLY ALGORITHM

The firefly algorithm is a SI technique [34] proposed by Yang in 2008 to simulate the luminous behavior of fireflies. In nature, fireflies transmit information through luminous signals. Inspired by this phenomenon, FA abstracts the luminescence and attraction characteristics of fireflies to establish the algorithm model. The main idea of the algorithm is that fireflies are attracted by other more attractive neighbors and move towards them, and finally converge to the optimal solution.

FA is based on the following three basic rules [35]: 1) Fireflies are unisex; 2) The attraction of fireflies is proportional to their brightness. In other words, the firefly with lower brightness will move to the other firefly that is brighter. The brightness is also affected by the distance between them, that is, the relative brightness will decrease as the distance increases. If there is no brighter firefly than the current one, it will move randomly in the search space. 3) The fitness value decides the brightness of the firefly. In the situation of solving the minimum problem, the smaller the fitness value of the firefly, the brighter it is and the more it can attract other fireflies towards him. Fireflies exchange information through brightness and attractiveness to find the best solution and eventually converge through this mechanism [36].

According to the design of FA, the brightness of a firefly is described by Eq. (3) as follows:

$$I_{oi} = f(X_i) \tag{3}$$

where X_i represents the ith firefly, and I_{oi} is the brightness of it. $f(X_i)$ represents the fitness value of X_i. As mentioned above, the relative brightness of each firefly in the vision field of other fireflies is also related to the distance between them, which can be described as Eq. (4).

$$I_i = I_{oi} \cdot e^{-\gamma \cdot r_{ij}} \tag{4}$$

where I_i is the relative brightness and γ represents the light absorption coefficient. r_{ij} is the distance between the ith firefly and the jth firefly, which can be defined as follows:

$$r_{ij} = ||X_i - X_j|| = \sqrt{\sum_{n=1}^{d} (x_{i,n} - x_{j,n})^2} \tag{5}$$

where d represents the dimension of the solution problem, $x_{i,n}$ and $x_{j,n}$ represent the nth location value of X_i and X_j respectively. Then the attraction of fireflies can be defined as:

$$\beta = \beta_0 \cdot e^{-\gamma \cdot r_{ij}^2} \tag{6}$$

where β_0 is the attraction when $r_{ij} = 0$. In the tth iteration, if the ith firefly is attracted by the jth firefly, its movement is calculated as shown in Eq. (7).

$$X_i^{t+1} = X_i^t + \beta \cdot \left(X_j^t - X_i^t\right) + \alpha \cdot (rand - 1/2) \tag{7}$$

where t represents the iteration number, α is the step parameter, and $rand$ is a random number uniformly distributed in [0,1]. In each iteration, Eq. (7) is used to execute the update process of each firefly circularly, and the algorithm will eventually converge to the optimal value. The pseudo-code of the FA is shown in **Algorithm 2**.

Algorithm 2 Pseudo-Code of FA

Input: The maximum number of iterations T, the population size N, the parameters α, β_0, γ
Output: The optimal solution that the algorithm found
Initialize a population of fireflies and evaluate the swarm by the fitness function.
For $t \leftarrow 1$ to T do
For $i \leftarrow 1$ to N do
For $j \leftarrow 1$ to N do
If $f(X_j) < f(X_i)$ then
Move the firefly X_i towards the firefly X_j using Eq. (7).
Update the fitness value of the new X_i.

C. EXTREME GRADIENT BOOSTING (XGBOOST)

The main premise of ensemble learning is that the error of a single classifier may be compensated by other classifiers when the model combines multiple models, which provide better performance than a single classifier [37]. Boosting is a technique that combines the weak classifiers (slightly better than random) into a stronger model in an iterative way [38]. XGBoost is one of the boosting technologies which utilizes trees as base learners. Specifically, it is a scalable tree boosting system. Since XGBoost was proposed by Chen and Guestrin in 2016 [39], it has attracted extensive attention and has been used in various competitions on the Kaggle machine learning competition website due to its high accuracy and excellent algorithm performance. XGBoost is a new method based on GBDT but differs in details such as loss functions [8]. The details of XGBoost are shown as follows:

Assume there is a dataset $D = \{(x_i, y_i)\}\,(i = 1, 2, \ldots, n)$ that contains n samples and m features, and k basic models are used to compose the additive model. Then the result $(\hat{y}_i)$ can be represented as follows:

$$\hat{y}_i = \emptyset\,(x_i) = \sum_k^K f_k\,(x_i), f_k \in F \tag{8}$$

$$F = \left\{f\,(x) = w_{q(x)}\right\}\left(q : \mathbb{R}^m \to T, w \in \mathbb{R}^T\right) \tag{9}$$

where F is the functional space and $f(x)$ is a regression tree. $q(x)$ represents the corresponding leaf node of xth sample, T is the number of leaves in the tree, and w is the leaf score. The predicted result of the t-th iteration can be described as follows:

$$\hat{y}_i^t = \hat{y}_i^{t-1} + f_t\,(x_i) \tag{10}$$

where $\hat{y}_i^{t-1}$ represents the predicted result of the instance i at iteration $t-1$. Then the objective function is defined as:

$$J\,(f_t) = \sum_{i=1}^n l\left(y_i, \hat{y}_i^{t-1} + f_t\,(x_i) + \Omega(f_t)\right) \tag{11}$$

where l is the training loss function. The regularization term $\Omega(f_t)$ represents the complexity of the model and it is calculated using Eq. (12):

$$\Omega\,(f) = \gamma \cdot T + \frac{1}{2}\lambda\,\|w\|^2 \tag{12}$$

where γ and λ can be used to control the complexity of the tree. By the way of using the regularization term, the complexity of the model can be controlled to avoid overfitting the training data.

IV. PROPOSED METHODS

A. REPRESENTATION OF AGENTS

SI algorithms are mostly designed for traditional continuous optimization problems. For optimization problems in different application problems, some components should be modified accordingly to make the algorithm better adapted to the solution of the problem. In the parameter tuning problem of the XGBoost algorithm that is mainly discussed in this paper, the range and meaning of each parameter are mostly different. Therefore, this paper adopts a more standardized representation of agents, which is convenient for algorithm interpretation and optimization. Assuming that k parameters of the XGBoost algorithm are chosen to be adjusted, we choose to use a proportional way to represent the value of parameters for the reason that each parameter has different requirements for the value range and value type. To be more specific, the proportional method makes use of the random number generated in [0,1] to represent the proportion of the value range of the parameter represented by this position. There are k parameters in total, so the length of the agent position is also k. The schematic diagram is shown in the figure (assuming $k = 5$):

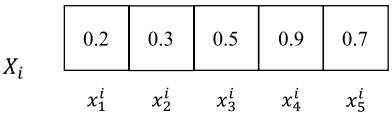

FIGURE 1. Representation of agents.

where X_i represents the ith agent. $x_j^i (j = 1, 2, 3, 4, 5)$ is the value of different parameter positions of the agent. When measuring the effect of the parameter combination represented by the agent, it is necessary to first convert the representation of the agent into available parameters before entering it into the model for calculation. The formula for this purpose is designed as follows:

$$param_j^i = left_j + \left(right_j - left_j\right) \cdot x_j^i \tag{13}$$

where $param_j^i$ represents the valid parameter value after conversion. $left_j$ and $right_j$ are the left and right bounds of the interval of values, respectively. With the above representation for agents, the SI algorithms can be applied to the problem of searching for the optimal combination of parameters of the XGBoost algorithm. The above content also facilitates the following elaboration about the proposed algorithm in this paper.

B. PROPOSED HYBRID FIREFLY AND BLACK HOLE ALGORITHM (HFBHA)

Aiming at solving the problem of parameter adjustment of the XGBoost algorithm, a hybrid algorithm that improves the BHA and integrates it with the FA is proposed in this paper. Firstly, a double black hole mechanism is designed to improve the update paths of the BHA, which adds the second black hole and attracts other stars at the same time. The update paths of stars are affected by two black holes, which helps the algorithm avoid falling into local optima due to the control of a single black hole. Secondly, the initialization method of the stars that are swallowed up by the black hole is improved by the proposed method. By this mechanism, the method explores more around the black hole and preserves the randomness, so as to improve the efficiency of the algorithm. Then, to increase the randomness of path selection and expand the global search scope, the FA is combined

as a perturbation strategy to supplement the updated paths of agents. Finally, the optimal agent mutation strategy is introduced to use the optimal solution of the current iteration for mutation, which is more convenient for the algorithm to search near the optimal solution.

1) DOUBLE BLACK HOLE MECHANISM

One of the drawbacks of traditional BHA is that the black hole has a great influence on the convergence process, which means the algorithm is easy to be manipulated by it. When the black hole is trapped in local optima, it is difficult for the algorithm to jump out of the trap because it still attracts the stars to move towards it. Therefore, this paper introduces the concept of the second black hole and considers the impact on other stars when there are double black holes in the population at the same time. The second black hole is the sub-optimal solution agent next to the current optimal solution (black hole) when sorting according to the fitness value. The optimal solution is still called the first black hole. If the first black hole and the second black hole simultaneously attract other stars to move towards it, the formula will be modified accordingly as shown in Eq. (14):

$$X_i^{t+1} = X_i^t + c_1 \cdot \left(BH_{First}{}^t - X_i^t\right) + c_2 \cdot \left(BH_{Second}{}^t - X_i^t\right) \tag{14}$$

where c_1 and c_2 represent the random number in [0,1]. BH_{First} and BH_{Second} represent the first black hole and the second black hole respectively. The meaning of this formula is that the stars of the t iteration are attracted by the first black hole and the second black hole at the same time and move a certain distance to each of them. The degree of movement is determined by the random number c_1 and c_2. Through the traction and balance of the second black hole relative to the first black hole, the motion trajectory of the stars will no longer tilt towards the optimal solution. The strategy makes it possible to explore more solutions in search space, improve the efficiency of the algorithm and avoid falling into local optima. For the reason that the calculation of the event horizon radius in the BHA is also related to the concept of the black hole, the initialization process is still based on the first black hole for simplicity. That is to say, the role of the second black hole only exists in star movements.

2) IMPROVED INITIALIZATION METHOD OF THE STARS

An important idea of the BHA is that a star too close to the black hole will be swallowed and the algorithm will generate a new star to replace it. The process of generating a new star in the traditional black hole algorithm is completely random. This random selection is equivalent to choosing a position in the search space completely at random without taking into account the priori information about the current optimal solution. It can be concluded that expecting the newly generated star to be close to the optimal solution is the same as the probability of generating an agent close to the optimal solution in a completely unknown search space. We believe that such a random way is less likely to provide effective information to help the search for an optimum. Based on this consideration, if the location of the current optimal solution can be utilized as the priori information to guide the generation of new stars, the local exploration capability of the algorithm and the computational resource utilization can be improved. The improved initialization strategy proposed in this paper is based on the assumption that the search near the location of current known optimal solutions can help generate stars with better quality than those generated by random initialization. It can be expected that if the new stars happen to be generated in the area of the global optimal solution, it will greatly simplify the computational effort required to converge. Specifically, the strategy will randomly generate new agents close to the current optimal solution (i.e., the first black hole). At the same time, to retain a certain degree of randomness, it is controlled using a threshold parameter, as mathematically defined in Eq. (15).

$$X_{New} = \begin{cases} BH_{First} + u \cdot V, \, if \; r_1 < \rho \\ X_{random}, \, else \end{cases} \tag{15}$$

where r_1 is a random number in [0,1]. ρ is used to control the probability of searching and generating stars near the black hole. u represents the random number generated from [-0.5,0.5] uniform distribution. V represents the generated vector with the same length as the first black hole. The values of each position of the vector V are random numbers in [0,1]. X_{random} represents the new star generated in a random way of the traditional BHA. Considering that if all the stars that cross the horizon radius are generated near the black hole, the algorithm may lose its original randomness. Therefore, the original random generation method has also been retained to a certain extent. The threshold parameter for controlling and balancing these two methods will help improve the performance of the algorithm.

3) FIREFLY PERTURBATION STRATEGY

Although the FA has the excellent searching ability, it has the problem of high complexity and slow convergence. In the FA, each iteration requires a double loop, which expands the search range but increases the computations required by the algorithm. Therefore, in order to take advantage of the FA to improve the efficiency of the BHA, this paper combines the FA with the proposed improved BHA and calls it a firefly perturbation strategy. In this strategy, there is a certain probability that the stars can be updated according to the trajectory of the fireflies. The firefly perturbation strategy improves the global exploration performance of the algorithm and provides the possibility of jumping out of local optima. Accordingly, in combination with the improvement of the BHA mentioned before, the movement formula for stars to be attracted by the black hole is defined as:

$$X_i^{t+1} = \begin{cases} X_i^t + c_1 \cdot \left(BH_{First}{}^t - X_i^t\right) + c_2 \cdot \left(BH_{Second}{}^t - X_i^t\right), \\ if \; r_2 < c \\ X_i^t + \beta_0 \cdot \left(X_j^t - X_i^t\right) + \alpha \cdot \left(rand - 1/2\right), \, else \end{cases} \tag{16}$$

where r_2 represents the random number in [0,1]. c is used to control the execution probability of the double black hole strategy and the firefly perturbation strategy. Its value needs to be carefully set and discussed. The meaning of each parameter in the second formula of Eq. (16) is the same as that of the FA described before. After adding the firefly perturbation strategy, the stars in the proposed method can select two paths to update, which is shown in FIGURE 2.

4) MUTATION OPERATOR

The mutation operator is a stochastic strategy that is widely used in SI algorithms. Adding the mutation operator to the proposed method in this paper will help the algorithm to jump out of the local optima. The mutation operation has less modification for the agent. Comparing the mutation of the non-optimal solution and the current optimal solution, the former may cost more modification than the latter. Therefore, if we take use of the current optimal solution for mutation, it may help to search for better solutions and approach the optimal solution. In summary, the mutation operator used in this paper focuses on the optimal solution of each iteration and preserves better results before and after the mutation. The main steps of the mutation operator are shown as follows:

Step 1: Select a position among the location of the current optimal agent at random.

Step 2: Generate a random number in [0,1].

Step 3: Replace the value at this position of the optimal agent with the random number selected in Step 2. Recalculate the fitness value of the agent.

Step 4: If the fitness value of the new agent is smaller than the original, replace it; Otherwise, it will not be replaced.

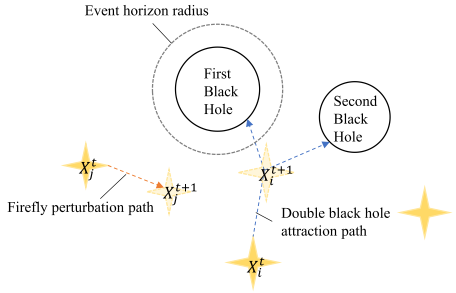

FIGURE 2. The update process of stars in the proposed method.

5) BRIEF SUMMARY

In conclusion, the algorithm proposed in this paper mainly optimizes the algorithm through the strategy mentioned above, and its pseudo-code is shown in **Algorithm 3**. The final optimal solution of the proposed method is the best parameter combination of the XGBoost algorithm. FIUGRE 3 shows the research flowchart of this paper.

Algorithm 3 Pseudo-Code of HFBHA

Input: The maximum number of iterations T, the population size N, the FA parameters α, β_0, γ, the controlling parameters ρ, c

Output: The optimal solution that the algorithm found

Initialize a population of stars.

Evaluate the swarm by the fitness function and set the best solution as the first black hole. The star that is only worse than the optimal solution is the second black hole.

For $t \leftarrow 1$ to T do
  For $i \leftarrow 1$ to $(N-1)$ do
    If random number $r_2 < c$ then
      Move the location of the star X_i using Eq. (14).
      Evaluate the fitness value of the star $f(X_i)$.
      If $f(X_i) > f(BH_{First})$ then
        Calculate the event horizon R using Eq. (2).
        If $\sqrt{(X_i - BH_{First})^2} < R$ then
          If random number $r_1 < \rho$ then
            Replace the current star X_i with a new star generated by Eq. (15).
          Else
            Replace the current star X_i with a new star generated randomly.
    Else
      For $i \leftarrow 1$ to N do
        Move the agent X_i towards the agent X_j using Eq. (16).
        If $f(X_i) < f(BH_{First})$ then
          Update BH_{First}, BH_{Second} and X_i.
    Perform the mutation operator on the optimal solution.

V. EXPERIMENTS AND RESULTS

In this section, the dataset used for experiments in this article is introduced. The main preprocessing steps are shown in detail. At the same time, the details of the experiment have been described, including the evaluation criteria, the experiment settings, and the sensitivity analysis. Finally, the elaborate results of the experiment are given and discussed.

A. DATASETS DESCRIPTION

Datasets play an important role in the research of IDSs [4]. At present, most of the research on intrusion detection technology is centered on open standard data sets [40], mainly KDD Cup 99 [41], NSL-KDD [42], UNSW-NB15 [43], CICCES-2017 [44], etc. Among them, the NSL-KDD and KDD Cup 99 datasets are widely welcomed by researchers because of their data availability. The NSL-KDD datasets are also used in this paper. One of the most important defects of the KDD Cup 99 dataset is a large number of redundant records. The redundancy will cause the machine learning algorithm to favor frequent records and make it more difficult to identify the patterns of infrequent records. NSL-KDD is an effort made by Tavallaee et al. [42] to solve the existing problems of the KDD CUP 99 dataset. The NSL-KDD dataset deletes all redundant instances, and gives a reasonable division of training and test datasets, providing a more perfect available intrusion detection dataset.

TABLE 1. The feature information of the NSL-KDD dataset.

Category	No.	Features	Type	Sample Data	Category	No.	Features	Type	Sample Data
Basic	1	duration	Numerical	0	Content	22	is_guest_login	Binary	0
	2	protocol_type	Nominal	tcp		23	count	Numerical	3
	3	service	Nominal	http		24	srv_count	Numerical	3
	4	flag	Nominal	SF		25	serror_rate	Numerical	0
	5	src_bytes	Numerical	233		26	srv_serror_rate	Numerical	0
	6	dst_bytes	Numerical	616		27	rerror_rate	Numerical	0
	7	land	Binary	0		28	srv_rerror_rate	Numerical	0
	8	wrong_fragment	Numerical	0		29	same_srv_rate	Numerical	1
	9	urgent	Numerical	0		30	diff_srv_rate	Numerical	0
Content	10	hot	Numerical	0		31	srv_diff_host_rate	Numerical	0
	11	num_failed_logins	Numerical	0	Traffic	32	dst_host_count	Numerical	66
	12	logged_in	Binary	1		33	dst_host_srv_count	Numerical	255
	13	num_compromised	Numerical	0		34	dst_host_same_srv_rate	Numerical	1
	14	root_shell	Binary	0		35	dst_host_diff_srv_rate	Numerical	0
	15	su_attempted	Binary	0		36	dst_host_same_src_port_rate	Numerical	0.02
	16	num_root	Numerical	0		37	dst_host_srv_diff_host_rate	Numerical	0.03
	17	num_file_creations	Numerical	0		38	dst_host_serror_rate	Numerical	0
	18	num_shells	Numerical	0		39	dst_host_srv_serror_rate	Numerical	0
	19	num_access_files	Numerical	0		40	dst_host_rerror_rate	Numerical	0.02
	20	num_outbound_cmds	Numerical	0		41	dst_host_srv_rerror_rate	Numerical	0
	21	is_host_login	Binary	0		42	class	Nominal	normal

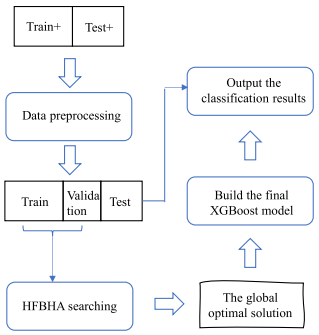

FIGURE 3. The flowchart of the research process.

NSL-KDD provides a set of downloadable files that can be used by researchers, which mainly includes two types of data files, namely ARFF and TXT file formats. In this set of files, KDD Train+ and KDD Test+ give the training set and test set divided by the authors. In most studies of IDSs, researchers usually choose the given 20% dataset [45] or generate their own randomly selected datasets [17], [18], [27], and rarely use the train and test sets given by the authors. Self-sampled datasets may have a bias or cannot represent the original dataset. Therefore, this paper attempts to give some guidance for the study by using Train+ and Test+ datasets. The feature information of the NSL-KDD dataset is shown in Table 1. As shown in Table 1, there are 41 features in this dataset, and only three feature data types are nominal. Normally, nominal data cannot be used in the machine learning model used in this paper, so it needs to be processed in the data preprocessing process. Table 2 shows the attack categories and mapping types of the training set Train+ and test set Test+ given in the NSL-KDD. The meaning of specific attack types is as follows [46]:

Denial of Service (**DoS**): In the DoS attack, the attacker sends a large number of requests to the server machine, resulting in its memory being too full or computing resources being too busy, thus denying service to the real user.

Probing Attack (**Probe**): This attack monitors and probes the network to obtain information, especially information about vulnerabilities, which can be later used to perform other types of attacks.

Remote to Local Attack (**R2L**): The attacker attempts to gain local access to the computer through the network from a remote computer without authorization.

User to Root Attack (**U2R**): The threat actor attempts to gain unauthorized access to the local superuser (root) or administrator privileges.

B. DATA PREPROCESSING

Based on the above analysis, it can be seen that there are different types of mapping attacks in KDD Train+ and KDD Test+. The total number of features of KDD Test+ is 15 more than that of KDD Train+, but there are 21 intersections of attack types in the two files. Therefore, this paper takes the common attack types of KDD Train+ and KDD Test+, then deletes the corresponding instances of 17 additional types in

TABLE 2. The feature information of the NSL-KDD.

Attack Class	Attack Type of Train+	Attack Type of Test+	Attack Type that both Train+ and Test+ have
DoS	Neptune, teardrop, smurf, pod, back, land	Neptune, teardrop, smurf, pod, back, land, apache2, processtable, worm, udpstorm, mailbomb	Neptune, teardrop, smurf, pod, back, land
Probe	ipsweep, portsweep, nmap, satan	ipsweep, portsweep, nmap, satan, saint, mscan	ipsweep, portsweep, nmap, satan
R2L	warezclient, guess_passwd, ftp_write, multihop, imap, warezmaster, phf, spy	guess_passwd, ftp_write, multihop, imap, warezmaster, phf, snmpgetattack, httptunnel, snmpguess, sendmail, xlock, xsnoop, named	guess_passwd, ftp_write, multihop, imap, warezmaster, phf
U2R	rootkit, buffer_overflow, loadmodule, perl	rootkit, buffer_overflow, loadmodule, perl, ps, xterm, sqlattack	rootkit, buffer_overflow, loadmodule, perl
Subtotal	**22**	**37**	**20**

the test set. To carry out the binary classification, the data labels are processed 0-1 instead of normal-abnormal.

It is noted that there are three nominal features (protocol_type, service, flag) in the dataset. In order to save the information of the original dataset as much as possible, we use the one-hot code for processing and then delete the original columns. The standardization of data helps to improve the accuracy and reduce the impact of features on model learning. Therefore, the dataset is also standardized in this paper.

During the one-hot coding process, a large number of new features are generated and the dataset becomes too large and too complex to use in the model training. Therefore, the principal component analysis (PCA) method for feature reduction is applied here. PCA is a technique that reduces the dimensionality of the dataset, increases interpretability, and minimizes information loss [47]. Using PCA techniques can help machine learning algorithms to learn hidden patterns in the dataset and reduce the noise and correlation of some features. In this paper, the percentage of principal component variance contribution obtained after applying the PCA technique is shown in FIUGRE 4. As shown in the figure, the color-filled part indicates the retained features that account for 99% of the components. In addition, 20% of the test set is sampled and selected as the validation set in this paper, i.e., the final dataset obtained is the training set of 125,081 instances, the test set of 100,064 instances, and the validation set is 25,017 instances, and they all contain 99 features.

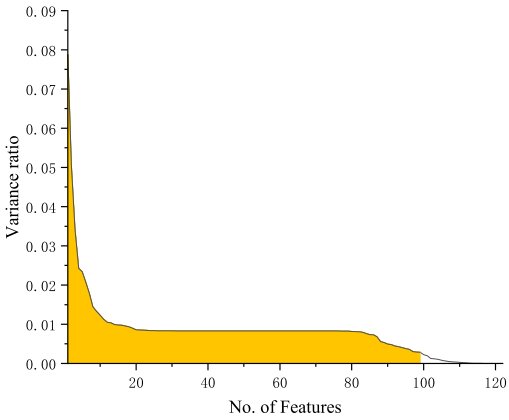

FIGURE 4. The principal component percentages.

C. EVALUATION CRITERIA

In this section, to evaluate the performance of the classifiers, we define a set of basic performance evaluation criteria. For classification tasks in machine learning, it is usually possible to use the output of the confusion matrix to calculate all selected measures [1]. The confusion matrix is represented by four main parameters, as follows:

True Positive (**TP**): TP represents the number of instances that are correctly classified as attacks.

False Positive (**FP**): FP represents the number of instances that are incorrectly classified as attacks. It is related to the first type of error.

True Negative (**TN**): TN represents the number of instances that are actually normal and correctly classified as normal.

False Negative (**FN**): FN represents the number of instances that are incorrectly classified as normal. It is related to the second type of error.

In this paper, FNR, FPR, F-score, and Accuracy are selected as evaluation criteria. The definitions of these indicators are as follows:

False-Negative Rate (**FNR**): The false negative rate, which is also known as the missed detection rate, indicates the number of samples that are misclassified as normal packets as a percentage of the number of all samples that are attacks. It can be defined as:

$$FNR = \frac{FN}{FN + TP} \tag{17}$$

False-Positive Rate (**FPR**): False positive rate, also known as false detection rate or false alarm rate, represents the proportion of the number of samples that are classified as attacks by mistake to the number of samples that are normal. The formula can be defined as:

$$FPR = \frac{FP}{FP + TN} \tag{18}$$

F-score: The F-score measures the accuracy of the model by considering the precision and recall rate at the same time, which is the harmonic average of them, as shown in Eq. (19):

$$F - score = \frac{2 \cdot TP}{2 \cdot TP + FN + FP} \tag{19}$$

Accuracy: It represents the proportion of the number of instances correctly classified to the total number of instances.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{20}$$

The missed detection rate and false alarm rate are important for the performance research of IDS. For IDSs, classifying normal behaviors as attacks may lead to problems such as wasting resources and influencing user experience. However, if the attack behavior is wrongly classified as normal behavior, it may have a huge and irreversible impact on the computer system and networks. Therefore, as indicated in the article [29], in order to detect any suspicious behavior in the network, a low missed detection rate is more important and meaningful than a low false detection rate.

In addition, a fitness function used for evaluating the quality of the model represented by the agents is also needed. The smaller the value of the fitness function, the better the performance of the model. The receiver operating characteristic (ROC) graph is a technology based on the performance visualization of classifiers to describe the trade-off between the hit rates and false alarm rates of classifiers [48]. For the aim of comparing the solutions, we may want to express ROC performance as a scalar value. A common method is to calculate the area under the ROC curve, namely area under curve (AUC). The use of AUC can judge the advantages and disadvantages of the classifier model. The value of AUC is within [0,1], and the higher value means the better the performance of the classifier. Therefore, the fitness function in this paper will be set as shown in Eq. (21):

$$Fitness_i = 1 - AUC_i \tag{21}$$

To compare the optimization ability of the heuristic algorithms and the performance of the model obtained by them, the minimum, maximum, average, and standard deviation of the fitness value will also be used as evaluation indicators for comparison and analysis.

D. EXPERIMENT SETTINGS

For the purpose of verifying the effectiveness of the HFBHA proposed in this paper, a series of comparative experiments are carried out. All experiments were executed in Anaconda 4.10.3 environment, running on AMD Ryzen 5 5600H with Radeon Graphics @ 3.30 GHz and 16.0GB RAM on the Windows 64-bit operating system.

In this paper, the original algorithms FA [34] and BHA [31], and the state-of-the-art algorithms QPSO [25], PSO [26], and WOA [27] are selected for comparison with the proposed algorithm in the experiments. The parameter settings of these SI algorithms are as suggested in the original articles.

At the same time, we also select the default XGBoost, Grid-search XGBoost, Grid-search Random Forest, and Grid-search LightGBM as the basic machine learning comparison algorithms. The SI algorithms will run 30 times independently, with a maximum number of iterations of 30 times and a population size of 15, and record the information about the optimal solution obtained. The non-heuristic traditional machine learning algorithm only needs to run once, because the obtained solution will be unique, and even if it runs 30 times, the minimum, maximum, and average values obtained are the same. Therefore, the machine learning algorithm is not included in Table 3 of the experimental results, but in Table 5.

During the operation of the SI algorithms, when calculating the fitness value of the agents, the value of the agent location is first converted into the value of the parameter combination. Then the parameter combination will be input into the XGBoost model to build the model. The training and verification sets are always used in the process of agent motion, while the test set will be retained in the model evaluation for final comparison. At the same time, all SI algorithms also search the five parameters of XGBoost model mentioned above, i.e., eta, gamma, max_depth, subsample, and colsample_bylevel. The meanings of these parameters are as follows:

eta: Reduce the weight of each step to prevent over-fitting. The default value is 0.3.

gamma: The minimum drop value of the loss function required for node splitting. The default value is 0.

max_depth: The maximum depth of the tree. The default value is 6.

subsample: Control the proportion of random samples per tree to prevent oversampling. The default value is 1.

colsample_bylevel: Control the percentage of columns that are randomly sampled per tree. The default value is 1.

E. SENSITIVITY ANALYSIS

As mentioned before, there are two parameters ρ and c that need to be set in the HFBHA algorithm proposed in this paper. Therefore, the settings of these two parameters will be analyzed in this section. The parameter ρ controls the probability of star regeneration around the black hole in the improved initialization method of the stars. In the sensitivity experiments, $\rho \in [0, 1]$, so ρ is set to take values in [0,1] in steps of 0.1. For the parameter c, it controls the choice of the two strategies during the agent's movement. If the right balance is achieved, the combination of these two strategies can be most effective. Therefore, the parameter c mainly affects the agent's motion trajectory, which means it has an important impact on the performance and complexity of the algorithm. In the proposed algorithm, the firefly perturbation strategy is an assisted strategy to improve the global search ability of the algorithm. If the parameter c tilts to the firefly perturbation strategy, it may cause the improvement strategy of the black hole algorithm to not fully play its role. Therefore, c is set to [0.6,0.7,0.8,0.9,1] here. When $c = 1$, the algorithm is completely transferred to BHA with the double black hole mechanism. For each combination of the two parameters mentioned above, the experiments run 20 times independently with the same settings. The average fitness value of all obtained optimal solutions for each parameter

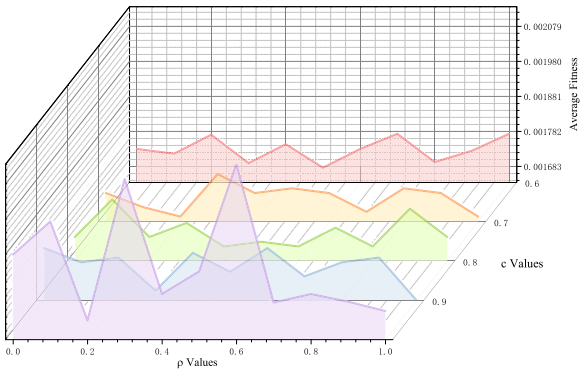

FIGURE 5. The average fitness value of parameter combination in sensitivity experiments.

combination is plotted in FIGURE 5 for the convenience of observing the pattern of parameter settings.

In FIUGRE 5, it can be observed that when $c = 1$, the curve shows that the degree of fluctuation in the influence of the value of ρ on the algorithm is large. However, with the decrease of the value of c, the fluctuation also decreases gradually. The possible reason for this phenomenon is that as the value of the parameter c decreases, the firefly perturbation strategy begins to participate in the algorithm, balancing the influence of ρ and improving the performance of the method. Comparing the curves for different values of the parameter c, it is observed that the algorithm performance is generally better when the value of c is chosen between 0.7 and 0.9. Although there is some volatility in the performance of the value for the parameter ρ, the value selection of [0.3, 0.7] tends to obtain relatively stable results under different c values. Considering the balance of the two strategies of the algorithm and the possible impact of time complexity, the experimental parameter settings of $c = 0.9$ and $\rho = 0.7$ are chosen in this paper.

F. EXPERIMENT RESULTS

This section will analyze and describe the comparison experiments conducted on the NSL-KDD dataset. Table 3 shows the experimental results of the SI algorithms after 30 runs. Among them, the minimum value in the comparison (i.e., the best optimal solution) has been marked in bold. It can be seen from Table 3 that the minimum, average and maximum values of HFBHA proposed in this paper are lower than other comparison methods, which shows that the performance of HFBHA is significantly better than other algorithms in experiments. Compared to the standard deviation of the SI algorithms, HFBHA has the lowest standard deviation, which indicates that the performance of this algorithm is stable. Combined with the above analysis, this algorithm can converge stably to the global optimal solution. The parameter combinations of the XGBoost algorithm found by the SI algorithms are shown in TABLE 4. FIGURE 6 shows the curve of the iterative process of the heuristic algorithms. It can

TABLE 3. The experiment results of the compared SI algorithms (10^{-3}).

Algorithms	Fitness(1-AUC)			
	Min	Mean	Max	Std
FA	1.5959	1.9329	3.8229	0.4100
BHA	1.9671	2.3092	2.6290	0.2348
QPSO	1.6702	1.8032	2.0351	0.1236
PSO	1.6516	1.8149	2.3878	0.1263
WOA	1.6454	1.8143	2.4744	0.1492
HFBHA	**1.5898**	**1.7516**	**1.9238**	**0.0774**

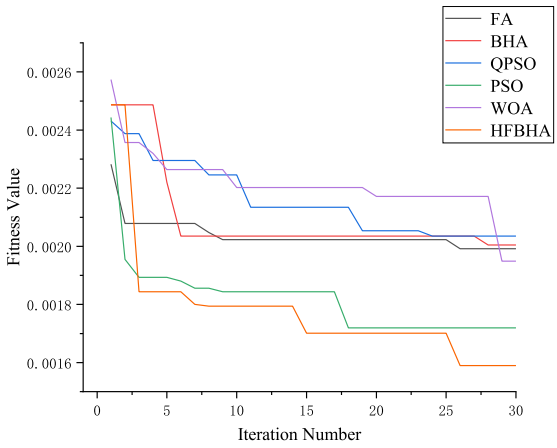

FIGURE 6. The iteration curves of the comparison algorithms.

TABLE 4. The parameter combinations found by the SI algorithms.

Algorithms	eta	gamma	max depth	subsample	colsample bylevel
FA	0.3062	8.3511	3	0.5721	0.2622
BHA	0.5360	9.8704	14	0.7170	0.7313
QPSO	0.6596	0	15	0.7704	0.1758
PSO	0.6608	0	15	0.7838	0.1726
WOA	0.6481	0	13	0.6521	0.9652
HFBHA	0.5766	0	18	0.7170	0.4802

TABLE 5. The experiment results of the comparison algorithms (10^{-2}).

Algorithms	FNR	FPR	F-score	Accuracy	AUC
FA	9.1068	20.6114	81.6853	83.9257	96.2041
BHA	9.0598	21.0902	81.2210	83.6011	96.6878
QPSO	7.8574	20.7079	81.9082	84.2609	96.1330
PSO	9.5027	20.5682	81.6247	83.8246	95.8585
WOA	7.8685	20.3036	82.3099	84.5482	96.7388
HFBHA	**3.7058**	**19.9322**	**83.7481**	**86.1019**	96.7663
XGBoost-Default	8.6914	20.9958	81.4082	83.7873	96.1371
XGBoost-GridSearch	8.9550	20.83	81.49	83.81	96.6492
RandomForest-GridSearch	9.2906	20.9701	81.2834	83.6117	**97.1232**
LightGBM-GridSearch	8.8907	20.8165	81.5378	83.8512	94.4776

be seen from FIUGRE 6 that the HFBHA proposed in this paper can converge at a faster speed and obtain better results when the initially generated optimal solutions are similar.

In addition to the SI algorithms, other ensemble learning algorithms are selected for comparison in this paper, using the evaluation criteria shown before. Table 5 shows the comparison results of the proposed algorithm, the SI algorithms,

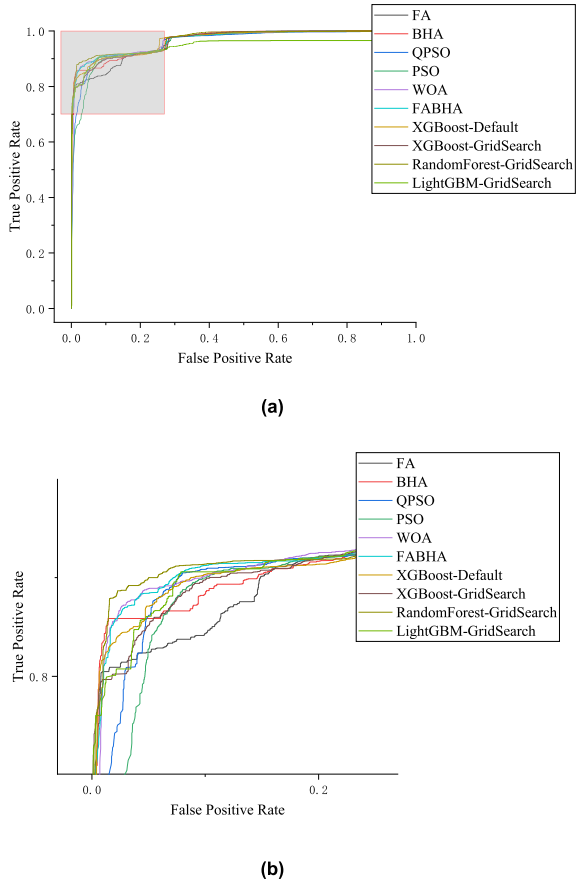

FIGURE 7. The ROC curves of comparison algorithms.

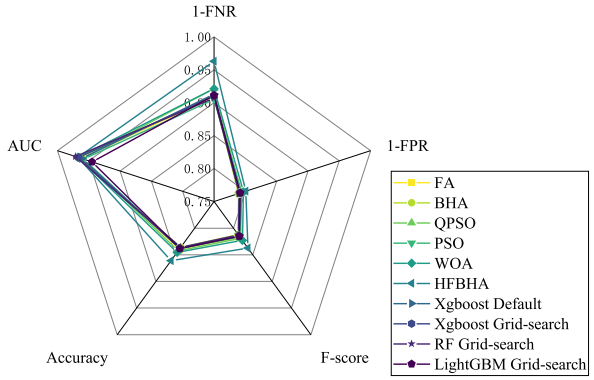

FIGURE 8. The radar plot comparing the experimental results of the algorithms.

and the ensemble learning algorithms. Since the heuristic algorithms were run several times, the performance of the best models they found is shown in Table 5. It is important to note that the calculation of fitness values of the algorithms, i.e., the values shown in Table 3, use the train and validation sets of this paper. However, the values in Table 5 are the results of the model performance evaluation using the test set. The FNR and FPR metrics represent the missed detection rate and false detection rate, respectively. It should be noticed that the lower values of these two values are better, while lower FNR is more important than lower FPR in IDS studies. Other indicators except FNR and FPR show that the larger the value, the better the model. Therefore, in Table 5, the best performance of this indicator in all algorithms has been marked in bold. It can be seen from the results that the performance of the HFBHA proposed in this paper outperforms the compared SI algorithms used in this experiment in all indicators. Comparing XGBoost using default values and grid-search, it can be found that the grid-search method has a certain effect on the tuning parameters of the XGBoost algorithm. However, the HFBHA is able to obtain a significant boost than the grid-search method. If we compare XGBoost, random forest, and LightGBM using Grid-search, it is obvious that Light-GBM performs better. HFBHA obtains greater results than the LightGBM using Grid-search on all metrics except AUC. The phenomenon that accuracy and AUC do not perform best at the same time may be attributed to the fact that they are not absolutely correlated. The ROC curves for all algorithms are plotted in FIUGRE 7, where the plot (b) is a partial enlargement of the plot (a). FIGURE 7 also shows that the ROC curve of LightGBM performs best, which indicates that there is still room for improvement in the proposed algorithm. The visualization of the experimental results is shown in FIGURE 8 using radar plots. In FIUGRE 8, the line that is mostly on the outermost side represents the proposed method. Combined with the above analysis, it can be seen that the performance of the HFBHA proposed in this paper outperforms the other selected comparison algorithms in most of the metrics except AUC. Although the proposed algorithm needs to be improved to some extent, this experiment is still able to prove the superiority of the algorithm, which is satisfactory in all other metrics. For the AUC values, HFBHA is slightly lower than LightGBM, but the gap between them is not huge. In summary, the experiments in this section verify the effectiveness of the HFBHA. The results analysis above also indicates that the proposed algorithm can take full advantage of tuning parameter combinations for the performance improvement of the XGBoost model.

VI. CONCLUSION AND FUTURE WORK

In this paper, we propose a hybrid firefly and black hole algorithm for the parameter tuning problem of the XGBoost model and apply it to the study of the IDS. The proposed algorithm first innovatively designs a double black hole mechanism to help the star move through the traction of two black holes, avoiding the problem that the algorithm is easily trapped in the local optimum. Secondly, an improved initialization method of the stars is introduced, which creatively takes into account the utilization of the prior information of the existing optimal solutions. Finally, the addition of the firefly perturbation strategy and mutation operators makes the global search performance of the algorithm improved and provides more possibilities to search for the global optimal

solution. This paper designs a kind of SI algorithm that automatically selects and optimizes the main parameters of XGBoost. This method can search a larger range of parameters, which has better performance than traditional manual setup or grid-search methods. The Train+ and Test+ datasets for NSL-KDD are analyzed and investigated. In this way, we can get out of the bias and generalization that can be introduced by traditional methods using datasets. Experimental results on the NSL-KDD dataset show that the model based on the algorithm proposed in this paper performs significantly better than the heuristic algorithms including FA, BHA, QPSO, PSO and WOA as well as the machine learning methods of default-XGBoost, grid-search XGBoost, grid-search random forest, and grid-search LightGBM. Therefore, the proposed classification method of HFBHA combined with XGBoost can be applied successfully to IDSs. In future research, the performance of HFBHA needs to be further improved. Besides, the algorithm proposed in this paper can be used for the parameter tuning problem of other models. Finally, the other intrusion detection datasets also deserve to be researched deeply.

REFERENCES

[1] M. Almseidin, M. Alzubi, S. Kovacs, and M. Alkasassbeh, "Evaluation of machine learning algorithms for intrusion detection system," Presented at the IEEE 15th Int. Symp. Intell. Syst. Inform. (SISY), Subotica, Serbia, 2017, doi: 10.1109/SISY.2017.8080566.

[2] Y. Zhou, G. Cheng, S. Jiang, and M. Dai, "Building an efficient intrusion detection system based on feature selection and ensemble classifier," *Comput. Netw.*, vol. 174, Jun. 2020, Art. no. 107247, doi: 10.1016/j.comnet.2020.107247.

[3] P. Mishra, V. Varadharajan, U. Tupakula, and E. S. Pilli, "A detailed investigation and analysis of using machine learning techniques for intrusion detection," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 1, pp. 686–728, 1st Quart., 2019, doi: 10.1109/COMST.2018.2847722.

[4] A. Khraisat, I. Gondal, P. Vamplew, and J. Kamruzzaman, "Survey of intrusion detection systems: Techniques, datasets and challenges," *Cybersecurity*, vol. 2, no. 1, pp. 1–22, Dec. 2019, doi: 10.1186/s42400-019-0038-7.

[5] Y. Rbah, M. Mahfoudi, Y. Balboul, M. Fattah, S. Mazer, M. Elbekkali, and B. Bernoussi, "Machine learning and deep learning methods for intrusion detection systems in IoMT: A survey," Presented at the 2nd Int. Conf. Innov. Res. Appl. Sci., Eng. Technol. (IRASET), Mar. 2022, doi: 10.1109/IRASET52964.2022.9738218.

[6] A. L. Buczak and E. Guven, "A survey of data mining and machine learning methods for cyber security intrusion detection," *IEEE Commun. Surv. Tuts.*, vol. 18, no. 2, pp. 1153–1176, 2nd Quart., 2016, doi: 10.1109/COMST.2015.2494502.

[7] M. I. Jordan and T. M. Mitchell, "Machine learning: Trends, perspectives, and prospects," *Science*, vol. 349, no. 6245, pp. 255–260, Jul. 2015, doi: 10.1126/science.aaa8415.

[8] C.-F. Tsai, Y.-F. Hsu, C.-Y. Lin, and W.-Y. Lin, "Intrusion detection by machine learning: A review," *Exp. Syst. Appl.*, vol. 36, no. 10, pp. 11994–12000, 2009, doi: 10.1016/j.eswa.2009.05.029.

[9] K. Song, F. Yan, T. Ding, L. Gao, and S. Lu, "A steel property optimization model based on the XGBoost algorithm and improved PSO," *Comput. Mater. Sci.*, vol. 174, Mar. 2020, Art. no. 109472, doi: 10.1016/j.commatsci.2019.109472.

[10] J. Nobre and R. F. Neves, "Combining principal component analysis, discrete wavelet transform and XGBoost to trade in the financial markets," *Exp. Syst. Appl.*, vol. 125, pp. 181–194, Jul. 2019, doi: 10.1016/j.eswa.2019.01.083.

[11] H. Li, Y. Cao, S. Li, J. Zhao, and Y. Sun, "XGBoost model and its application to personal credit evaluation," *IEEE Intell. Syst.*, vol. 35, no. 3, pp. 52–61, May 2020, doi: 10.1109/MIS.2020.2972533.

[12] Y. Jiang, G. Tong, H. Yin, and N. Xiong, "A pedestrian detection method based on genetic algorithm for optimize XGBoost training parameters," *IEEE Access*, vol. 7, pp. 118310–118321, 2019, doi: 10.1109/ACCESS.2019.2936454.

[13] Y. Qiu, J. Zhou, M. Khandelwal, H. Yang, P. Yang, and C. Li, "Performance evaluation of hybrid WOA-XGBoost, GWO-XGBoost and BO-XGBoost models to predict blast-induced ground vibration," *Eng. Comput.*, vol. 38, no. S5, pp. 4145–4162, Dec. 2022, doi: 10.1007/s00366-021-01393-9.

[14] M. H. Nasir, S. A. Khan, M. M. Khan, and M. Fatima, "Swarm intelligence inspired intrusion detection systems—A systematic literature review," *Comput. Netw.*, vol. 205, Mar. 2022, Art. no. 108708, doi: 10.1016/j.comnet.2021.108708.

[15] H. Alazzam, A. Sharieh, and K. E. Sabri, "A feature selection algorithm for intrusion detection system based on pigeon inspired optimizer," *Exp. Syst. Appl.*, vol. 148, Jun. 2020, Art. no. 113249, doi: 10.1016/j.eswa.2020.113249.

[16] B. Selvakumar and K. Muneeswaran, "Firefly algorithm based feature selection for network intrusion detection," *Comput. Secur.*, vol. 81, pp. 148–155, Mar. 2019, doi: 10.1016/j.cose.2018.11.005.

[17] A. C. Enache and V. V. Patriciu, "Intrusions detection based on support vector machine optimized with swarm intelligence," Presented at the 9th IEEE Int. Symp. Appl. Comput. Intell. Inform. (SACI), Timisoara, Romania, May 2014, doi: 10.1109/SACI.2014.6840052.

[18] A. C. Enache and V. Sgârciu, "Anomaly intrusions detection based on support vector machines with bat algorithm," Presented at the 18th Int. Conf. Syst. Theory. Control Comput. (ICSTCC), Sinaia, Romania, 2014, doi: 10.1109/ICSTCC.2014.6982526.

[19] F. Kuang, S. Zhang, Z. Jin, and W. Xu, "A novel SVM by combining kernel principal component analysis and improved chaotic particle swarm optimization for intrusion detection," *Soft Comput.*, vol. 19, no. 5, pp. 1187–1199, May 2015, doi: 10.1007/s00500-014-1332-7.

[20] J. Li, H. Wang, and B. Yan, "Application of velocity adaptive shuffled frog leaping bat algorithm in ICS intrusion detection," Presented at the 29th Chin. Control Decis. Conf. (CCDC), Chongqing, 2017, doi: 10.1109/CCDC.2017.7979135.

[21] M. H. Ali, M. Fadlizolkipi, A. Firdaus, and N. Z. Khidzir, "A hybrid particle swarm optimization -Extreme learning machine approach for intrusion detection system," Presented at the IEEE Student Conf. Res. Develop. (SCOReD), Nov. 2018, doi: 10.1109/SCORED.2018.8711287.

[22] H. Yu, M. Jia, X. Cheng, and Q. Jiang, "Optimized k-means clustering algorithm based on artificial fish swarm," Presented at the Int. Conf. Mech. Sci., Electric Eng. Comput. (MEC), Dec. 2013, doi: 10.1109/MEC.2013.6885342.

[23] P. Wei, Y. Li, Z. Zhang, T. Hu, Z. Li, and D. Liu, "An optimization method for intrusion detection classification model based on deep belief network," *IEEE Access*, vol. 7, pp. 87593–87605, 2019, doi: 10.1109/ACCESS.2019.2925828.

[24] A.-U.-H. Qureshi, H. Larijani, A. Javed, N. Mtetwa, and J. Ahmad, "Intrusion detection using swarm intelligence," in *Proc. U.K./China Emerg. Technol. (UCET)*, Glasgow, U.K., Aug. 2019, pp. 1–5, doi: 10.1109/UCET.2019.8881840.

[25] J. Wang, C. Liu, X. Shu, H. Jiang, X. Yu, J. Wang, and W. Wang, "Network intrusion detection based on XGBoost model improved by quantum-behaved particle swarm optimization," Presented at the IEEE Sustain. Power Energy Conf. (iSPEC), Beijing, China, Nov. 2019, doi: 10.1109/iSPEC48194.2019.8975295.

[26] H. Jiang, Z. He, G. Ye, and H. Zhang, "Network intrusion detection based on PSO-XGBoost model," *IEEE Access*, vol. 8, pp. 58392–58401, 2020, doi: 10.1109/ACCESS.2020.2982418.

[27] S. Cheng, Y. Song, H. W. Li, and D. Liu, "A method of intrusion detection based on WOA-XGBoost algorithm," *Discrete Dyn. Nature Soc.*, vol. 2022, Feb. 2022, Art. no. 5245622, doi: 10.1155/2022/5245622.

[28] M. Zivkovic, M. Tair, K. Venkatachalam, N. Bacanin, Š. Hubálovský, and P. Trojovský, "Novel hybrid firefly algorithm: An application to enhance XGBoost tuning for intrusion detection classification," *PeerJ Comput. Sci.*, vol. 8, p. e956, Apr. 2022, doi: 10.7717/peerj-cs.956.

[29] P. A. A. Resende and A. C. Drummond, "A survey of random forest based methods for intrusion detection systems," *ACM Comput. Surveys*, vol. 51, no. 3, pp. 1–36, May 2019, doi: 10.1145/3178582.

[30] M. M. Mafarja and S. Mirjalili, "Hybrid whale optimization algorithm with simulated annealing for feature selection," *Neuro Comput.*, vol. 260, pp. 302–312, Oct. 2017, doi: 10.1016/j.neucom.2017.04.053.

[31] A. Hatamlou, "Black hole: A new heuristic optimization approach for data clustering," *Inf. Sci.*, vol. 222, pp. 175–184, Feb. 2012, doi: 10.1016/j.ins.2012.08.023.
[32] R. Soto, B. Crawford, R. Olivares, C. Taramasco, I. Figueroa, Á. Gómez, C. Castro, and F. Paredes, "Adaptive black hole algorithm for solving the set covering problem," *Math. Problems Eng.*, vol. 2018, Oct. 2018, Art. no. 2183214, doi: 10.1155/2018/2183214.
[33] S. Kumar, D. Datta, and S. K. Singh, "Black hole algorithm and its applications," Presented at the Comput. Intell. Appl. Model. Control. Stud. Comput. Intell., vol. 575. Cham, Switzerland: Springer, 2015, doi: 10.1007/978-3-319-11017-2_7.
[34] X. S. Yang, "Firefly algorithms for multimodal optimization stochastic algorithms: foundations and applications," in *Stochastic Algorithms: Foundations and Applications* (Lecture Notes in Computer Science), Berlin, Germany: Springer, 2009, p. 5792.
[35] V. Kumar and D. Kumar, "A systematic review on firefly algorithm: Past, present, and future," *Arch. Comput. Methods Eng.*, vol. 28, pp. 3269–3291, Jun. 2021, doi: 10.1007/s11831-020-09498-y.
[36] X. Yong and Y.-L. Gao, "Improved firefly algorithm for feature selection with the ReliefF-based initialization and the weighted voting mechanism," *Neural Comput. Appl.*, vol. 35, no. 1, pp. 275–301, Jan. 2023, doi: 10.1007/s00521-022-07755-8.
[37] O. Sagi and L. Rokach, "Ensemble learning: A survey," *WIREs Data Mining Knowl. Discover* vol. 8, no. 4, p. e1249, 2018, doi: 10.1002/widm.1249.
[38] Y. Freund, R. E. Schapire, and N. Abe, "A short introduction to boosting," *J. Japn. Soc. Artif. Intell.*, vol. 14, no. 5, pp. 771–780, 1999.
[39] T. Q. Chen and G. Carlos, "XGBoost: A scalable tree boosting system," Presented at the 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining, 2016, pp. 785–794, doi: 10.1145/2939672.2939785.
[40] Z. Ahmad, A. S. Khan, S. Wai, J. Abdullah, and F. Ahmad, "Network intrusion detection system: A systematic study of machine learning and deep learning approaches," *Trans. Emerg. Telecommun. Technol.*, vol. 32, p. e4150, Jan. 2021, doi: 10.1002/ett.4150.
[41] (Oct. 2007). *KDD Cup 1999*. [Online]. Available: http://kdd.ics.uci.edu/databases/kddcu99/kddcup99.html
[42] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A detailed analysis of the KDD CUP 99 data set," Presented at the IEEE Symp. Comput. Intell. Secur. Defense Appl., Jul. 2009, doi: 10.1109/CISDA.2009.5356528.
[43] N. Moustafa and J. Slay, "UNSW-NB15: A comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set)," Presented at the Mil. Commun. Inf. Syst. Conf. (MilCIS), Canberra, ACT, Australia, Nov. 2015, doi: 10.1109/MilCIS.2015.7348942.
[44] S. Iman, H. L. Arash, and G. Ali, "Toward generating a new intrusion detection dataset and intrusion traffic characterization," Presented at the Int. Conf. Inf. Syst. Secur. Privacy, 2018, doi: 10.5220/0006639801080116.
[45] L. Dhanabal and S. P. Shantharajah, "A study on NSL-KDD dataset for intrusion detection system based on classification algorithms," *Int. J. Adv. Res. Comput. Commun. Eng.*, vol. 4, no. 6, pp. 446–452, 2015, doi: 10.17148/IJARCCE.2015.4696.
[46] R. Thomas and D. Pavithran, "A survey of intrusion detection models based on NSL-KDD data set," Presented at the 5th HCT Inf. Technol. Trends (ITT), Dubai, United Arab Emirates, Nov. 2018, doi: 10.1109/CTIT.2018.8649498.
[47] I. T. Jolliffe and J. Cadima, "Principal component analysis: A review and recent developments," *Phil. Trans. Roy. Soc. A, Math., Phys. Eng. Sci.*, vol. 374, no. 2065, Apr. 2016, Art. no. 20150202, doi: 10.1098/rsta.2015.0202.
[48] T. Fawcett, "An introduction to ROC analysis," *Pattern Recognit. Lett.*, vol. 27, no. 8, pp. 861–874, Jun. 2006, doi: 10.1016/j.patrec.2005.10.010.

XIN YONG received the B.Sc. degree from the Shanghai University of Finance and Economics, China, in 2020. She is currently pursuing the M.Sc. degree with North Minzu University. Her research interests include optimization theory, machine learning, and intrusion detection.

YUELIN GAO received the B.Sc. degree from Yan'an University, Yan'an, China, in 1984, the M.Sc. degree from the Dalian University of Technology, Dalian, China, in 1991, and the Ph.D. degree from Xi'an Jiaotong University, Xi'an, China, in 2002. He is currently a Full Professor and a Ph.D. Supervisor with North Minzu University, Yinchuan, China. He has published more than 150 academic articles in important journals. His current research interests include global optimization, evolutionary computing, optimization theory and method, and financial statistics.