A Combinatory Feature Selection Method using Gray Wolf Optimization and Crow Search Algorithms for Intrusion Detection Systems

Keyvan Asghari
Department of Computer Engineering, Sardroud Center, Tabriz Branch, Islamic Azad University, Tabriz, Iran
Email:dr.asghari@iau.ac.ir
Receive Date: 10 September 2022, Revise Date: 6 October 2022, Accept Date:20 November 2022

Abstract

The crow search algorithm and the grey wolf optimizer are two favored optimization approaches that have attracted extensive attention from researchers. The swarm-based lifestyle of crows in nature is the inspiration source of the crow search algorithm. The grey wolf optimizer is inspired by the hierarchical system of grey wolves for hunting. Both mentioned algorithms perform properly for solving many optimization problems, but these algorithms do not perform well for some. A combinatory optimization algorithm is introduced in this paper by combining the crow search algorithm with the grey wolf optimizer. The introduced approach has more diverse movements to explore the search space of the investigated problem. The combinatory algorithm is used to solve the feature selection problem of intrusion detection systems, where its goal is to improve the accuracy rate by selecting the most important features to build the system's classifier. The UNSW-NB15 intrusion detection dataset is considered for evaluation of the combinatory algorithm. The results of the experiments reveal the high efficiency of the combinatory algorithm for most instances in the experiments in comparison with the other popular optimization algorithms.

Keywords: Gray wolf optimization, Crow search algorithm, Intrusion detection, Feature selection, Combinatory optimization algorithm.

1- Introduction

Regardless of their nature, swarm-based optimization algorithms share a common feature of establishing the search process from the exploration and exploitation phases. Determining a proper balance between these two steps is the most important mission in creating an optimization algorithm. Along the exploration step, the algorithm attempts to search the complete search space of a problem and with long movements. In the exploitation step, the algorithm searches the search space close to the fortunate answers found in the exploration phase with small moves.

One of the problems of optimization algorithms is that the algorithm stops at local optimal answers where the answers do not improve despite more iterations. In some cases, the problem of stagnation in the answers arises, where after some iterations, the algorithm does not produce new and better answers. In this case, obtaining the global optimal answer is not possible. The above two problems usually arise due to the poor performance of the algorithm in the exploration phase, which is usually solved by increasing the diversity in the population by adding new operators. One of the other problems of optimization algorithms occurs in the exploitation phase. When suitable potential answers have been found, it is not desirable to continue the search and exploration operations in the search space,

and the optimal global answer should be searched for by local search around the found answers. Optimization algorithms are usually equipped with parameters that control the portion of exploration and exploitation to solve the problem. These parameters try to direct the search in the problem search space as completely as possible in the initial iterations of the algorithm. But in the final iterations of the algorithm, the mentioned parameters reduce the power of exploration and try to focus the algorithm on the answers found in the exploration phase and more accurate local search around these answers. In general, creating a reasonable proportion between the exploration and exploitation function of the algorithm will cause the algorithm to converge gradually to answers close to the global optimum answer. Another approach suggested in the literature to solve the above issues is to combine optimization methods to use their benefits and conceal the weaknesses of each method.

The number of intrusions and attacks in computer systems has increased with the growing network connections and the use of different network-based software. Intrusion detection systems are presented as a proper technology for this issue and are considered an essential component of computer networks. Intrusion detection systems have different types based on the kind of information detection and analysis them. They can be organized as anomaly-based systems and abuse-based systems. Designing an intrusion detection system contains two main steps for selecting features and creating an optimal event classifier employing the selected features. Various optimization approaches have been used for feature selection in intrusion detection systems[1]–[7].

An optimization method combining the gray wolf and crow search algorithms for feature selection in intrusion detection systems is introduced in this paper. The used classifier is a naive Bayesian network. The combination of different operators of optimization algorithms has increased the power of exploration and exploitation and caused to find better outcomes by the proposed combinatory algorithm.

2- Previous works

Various optimization algorithms have been proposed so far to solve diverse optimization problems. The gray wolf optimization [8] and crow search [9] algorithms are two of them that have attracted the attention of several researchers. In this section, a brief explanation of the structure and functionality of the mentioned algorithms is provided.

2-1- The gray wolf optimization

The Gray Wolf Optimization algorithm, which is a swarm-based meta-heuristic, is introduced by Mirjalili et al. and simulates the hierarchical leadership of gray wolves for hunting in nature[8]. The alpha, beta, delta, and omega are the wolf types in the gray wolf algorithm which includes the three phases of the optimization process: encircle the prey, search for it, and attack it. Alpha is the leader wolf who directs other wolves for hunting. There are beta wolves in the second level, which provide communication between the alpha and delta wolves. The delta and omega wolves place in the third and fourth levels of the hierarchy of gray wolves.The distance of the current wolf (W) from alpha, beta, and delta wolves must be computed with Equation (1) for each iteration to obtain its next position in the search space.

$$\begin{aligned} D_{alpha} &= |C_1 . W_{alpha} - W| \\ D_{beta} &= |C_2 . W_{beta} - W| \\ D_{delta} &= |C_3 . W_{delta} - W| \end{aligned} \quad (1)$$

Equation (2) calculates *W1*, *W2*, and *W3* for obtaining the next position of the *W* wolf.

$$\begin{aligned} W_1 &= W_{alpha} - A_1 \,.\, D_{alpha} \\ W_2 &= W_{beta} - A_2 \,.\, D_{beta} \\ W_3 &= W_{delta} - A_3 \,.\, D_{delta} \end{aligned} \quad (2)$$

Equation (3) computes the control parameters of the algorithm named A and C, where a is a number between 2 and 0 and decreases linearly along the iterations. Two randomly generated values, r_1 and r_2, are between 0 and 1 in Equation (3).

$$\begin{aligned} A &= 2\,a.\,r_1 - a \\ C &= 2\,.\,r_2 \end{aligned} \quad (3)$$

The new position of each wolf in the population for use in the next iteration is calculated by Equation (4).

$$W(t+1) = (W_1 + W_2 + W_3)/3 \quad (4)$$

The gray wolf optimization algorithm has been applied for various applications, and its modified versions have also been developed.

2-2- The Crow Search Optimization

Crows are known to hide their food to use later, and simultaneously they watch other birds for finding and stealing the foods they have hidden. In the crow search algorithm [9], a group of crows with the size of *N* are interacting in an environment with the dimension of *d*. The crow with the number *a* has a position in the search space for each iteration which is indicated by $x_{a,iter}$. The crow *a* memorizes its best food-hiding place in the position $m_{a,iter}$. Crows update their hiding places by moving in the search space and finding better places, which will be a food source in the future. In the iterations of the crow search algorithm, when the crow *b* wants to find its hiding place $m_{b,iter,}$ and crow *a* wants to follow crow *b* to find its hiding place, two situations may happen.

In the first situation, crow *b* does not know that crow *a* aims to follow it. Therefore, crow *a* will find the hiding position of crow *b* and the new position of crow a is calculated with Equation (5), where r_i is a random value in [0, 1] and $fl_{a,iter}$ is the flight length of crow *a* for the iteration *iter*. $AP_{b,iter}$ is the awareness probability of crow *b* for iteration iter. The local search near the $x_{a,iter}$ is performed by small amounts for the $fl_{a,iter}$ and the global search is done by large amounts for it.

Crow *a* follows crow *b* in the second situation, while crow *b* knows about that. Therefore, crow *b* misleads crow *a* by flying to another position to protect its food hiding position. For the second situation, the new position of crow a is obtained by giving it a random position of the search space (Equation (6)).

$$x_{a,iter+1} = x_{a,iter} + r_a \times fl_{a,iter} \times (m_{b,iter} - x_{a,iter}) \quad r_b \geq AP_{b,iter} \quad (5)$$

$$x_{a,iter+1} = a\ random\ position \quad r_b < AP_{b,iter} \quad (6)$$

3- Combinatory Crow Search and Gray Wolf Algorithm

In this section, using the crow search and gray wolf optimization algorithms, a combinatory optimization approach is proposed to solve the optimization problem of feature selection in intrusion detection systems, which takes advantage of both algorithms. Like the other optimization algorithms, crow search and gray wolf optimization may get trapped in local optima and find low-quality answers due to fast convergence. The proposed combinatory algorithm improves the exploration and exploitation phases by having a variety of motion operators related to both algorithms for the search elements. It creates a balance between these two phases and can find the near-optimal global answer. It makes an appropriate balance between these two phases

where it can find the near-optimal global answer. The proposed combinatory algorithm efficiently explores the studied problem's search space and finds promising regions. The reason for the high exploration property of the combinatory algorithm is the existence of search elements with different answers from its base algorithms. In the later iterations of the combinatory algorithm, the operators of it, inherited from the crow search and gray wolf, perform a more detailed search near the promising answers found in the exploration phase of the algorithm. The combinatory algorithm appropriately balances the exploration and exploitation steps, and by finding accurate answers, its performance is higher than the crow search and gray wolf algorithms. The most important properties of the introduced combinatory algorithm are as follows:

- The proposed algorithm is developed by combining two optimization algorithms, the crow search and gray wolf optimization, so it has more diverse movement operators for its search elements in the problem search space.
- The combinatory algorithm uses random values within the range of the search space to initialize the search elements. A random population is generated in the initialization phase of the combinatory algorithm. This population is divided into two subpopulations. The crow search algorithm works on the former, and the gray wolf algorithm works on the latter.
- In each iteration of the combinatory algorithm, the two algorithms of the crow search and the gray wolf optimization work on two independent populations, and at the end of each iteration, the populations are incorporated as a single merged population.
- Subsequently, the fitness of the members of the merged population is calculated and sorted, and a fair share of search elements based on the fitness value is divided between the crow search and gray wolf populations. For this purpose, after sorting, members with even numbers are considered for search elements of the crow search algorithm, and members with odd numbers are considered for search elements of the gray wolf algorithm.
- In the early iterations of the combinatory algorithm, the search elements are moving in the search space by different strategies of the crow search and gray wolf algorithms, where the initial population is generated completely randomly. Nevertheless, in the last iterations, the coefficients of both base algorithms cause to decrease in the diversity of search elements, and the population converges to the near-optimal answers. The mentioned behavior change of the algorithm establishes a balance in the proportion of exploration and exploitation operations.

The flowchart of the combinatory optimization algorithm is illustrated in Figure 1. According to Figure 1, the algorithm has two-population that applies a different algorithm to each sub-population and improves the population elements. The important specialty of the combinatory algorithm is that in the current state, the operators of the crow search and gray wolf algorithms are applied sequentially on the sub-populations, and due to the halving of the population, there is no considerable increase in the algorithm execution time compared to the base algorithms. But due to the independence of the sub-populations of the proposed combinatory algorithm, it is possible to run the crow search and gray wolf algorithms in parallel on the relevant sub-populations, and merging and division of sub-populations are performed only at the end of each iteration.

The execution time of the combinatory algorithm will be reduced to about half as a result. In the proposed algorithm, an initial population is generated first randomly. In the next step, the proposed algorithm checks random answers. If they are outside the problem search space, the answers will be returned to the search space. Afterward, the combinatory algorithm computes the fitness values for the randomly generated answers. Then, the algorithm partitions the merged population into two parts considering the fitness values of search elements. This process establishes equal conditions for both crow search and gray wolf parts to improve and update the sub-populations.

The pseudo-code of the introduced combinatory grey wolf-crow optimization method is presented in Algorithm 1.

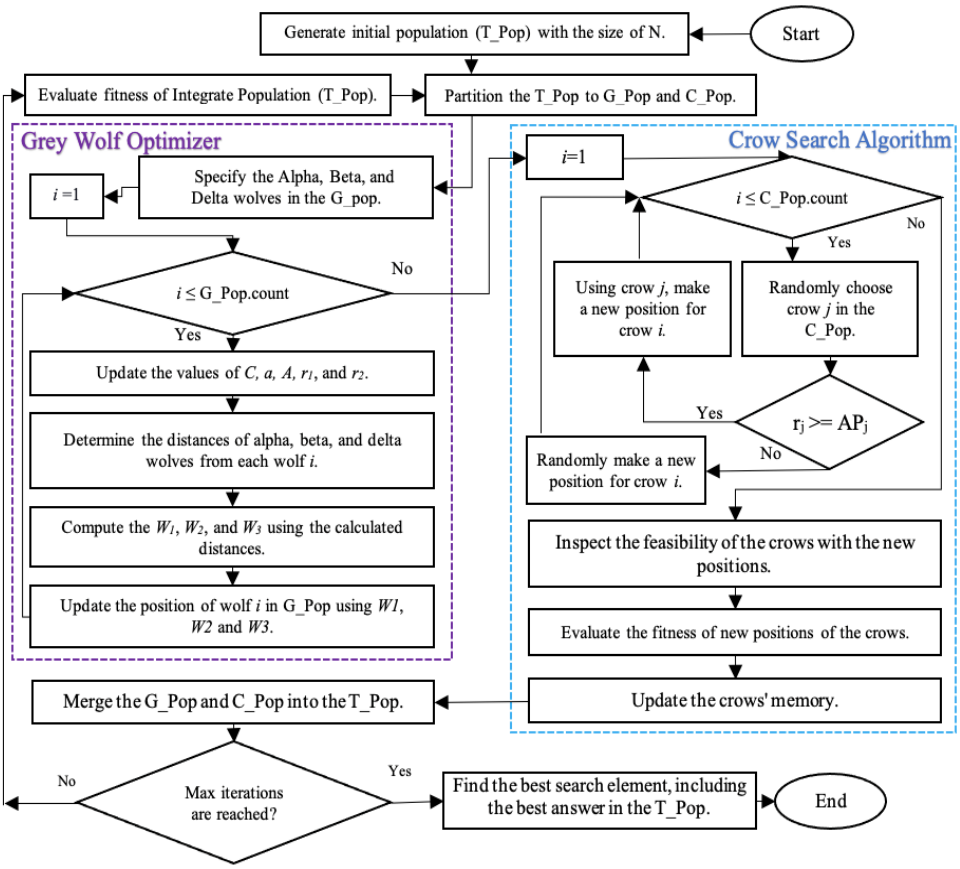

Fig. 1. Flowchart of the combinatory gray wolf-crow optimization algorithm

Algorithm 1. Pseudo-code for the combinatory grey wolf-crow optimization algorithm

T_Pop = Create a population randomly with the size of N (Size of the total population);
While (i < max iterations)
  Check the feasibility of answers in the T_Pop;
  Compute the T_Pop's fitness values;
  Partition the T_Pop to G_Pop (wolves, GW_j (j=1,2, …, N/2)) and C_Pop (crows, CR_j (j = 1, 2, ..., N/2));
  Determine the Alpha, Beta, and Delta wolves in G_Pop.;
  For each GW_j (gray wolf) in G_Pop (j = 1, 2, ..., N/2)
    Compute C, A, a, W1, W2, and W3 by Equation (2) and Equation (3);
    Compute the new position of the GW_j using W1, W2, and W3 and Equation (4);
  End For
  Initialize the memory of each crow in C_Pop;
  For each CR_k in C_Pop (k = 1, 2, ..., N/2)
    CR_j = Select a crow randomly in C_Pop to follow it;
    AP_j = Define an awareness probability;
    If r_j >= AP_j

```
        CRk.new_position=Change the position of the CRk by Equation (5);
      Else
        CRk.new_position=Change the position of the CRk by Equation (6);
      End If
    End For
    Check the feasibility of new positions of the crows in C_Pop;
    Evaluate the fitness of new positions of the crows in C_Pop;
    Update the crows' memory.
    Integrate G_Pop (wolves) and C_Pop (crows) into the T_Pop;
    i=i+1;
  End While
  Return the best-answer in the T_Pop;
```

3-1- Using the proposed algorithm for feature selection

There are many features in every networking event, and investigating them is a time-consuming job. On the other hand, some features are duplicated and provide no new information about the network event. Therefore, some feature reduction techniques are introduced so far that decrease the complexity of this problem[3]. An optimal feature set is selected to build an event classifier in intrusion detection systems. This set will guarantee the speed and accuracy of the classifier by representing all features. In an intrusion detection system, the job function of a classifier is categorizing the network packets (events) into normal and attacks.

The introduced combinatory method is evaluated by optimizing the selected set to build the classifier in the current paper. The naive Bayesian network [10] as a simple and easy-to-use classifier is applied in the experiments of this paper. At the end of each iteration of the compared algorithms, a naive Bayesian network is built by the selected features and trained with the intrusion detection dataset. Afterward, the classifier is evaluated with the test set of the dataset. The naive Bayesian network has many other applications as a data mining tool which classification of network events in intrusion detection systems is one of them. There are different kinds of classifiers that are used in intrusion detection systems. However, this paper focuses on the feature reduction phase of developing an intrusion detection system. Figure 2 shows the flowchart of using the combinatory algorithm for the feature selection phase of developing an intrusion detection system.

4- Evaluation Results of the Combinatory Algorithm for Feature Selection

The UNSW-NB15 dataset [11] is used to conduct experiments and evaluate the performance of the feature selection algorithm presented in this paper. The existing and proposed algorithms have been implemented by MATLAB software, and the obtained results have been compared with previous methods.

4-1- The UNSW-NB15 dataset

The dataset used in this paper's experiments, which performs better in detecting newer attacks, is the UNSW-NB15 dataset [11]. There are 42 features in the records of the UNSW-NB15 dataset, where the attacks have nine types. Table 1 indicates the contents of the UNSW-NB15 dataset records.

Table 1. The types and number of training and test records in the UNSW-NB15 dataset

Record Kind	Train Record	Test Record
Analysis	2000	677
Exploits	33393	11132
Fuzzers	18184	6062
Shellcode	1133	378
Reconnaissanc	10491	3496
Generic	18871	40000
Backdoors	1746	583
DoS	12264	4089
Worms	130	44
Normal	56000	37000
Total Records	**175341**	**82332**

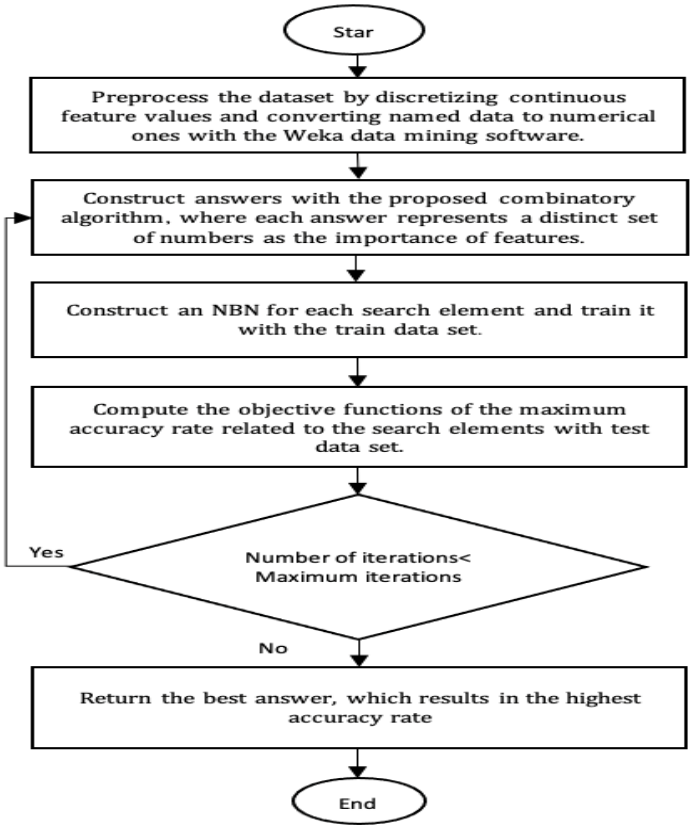

Fig 2. Using the combinatory algorithm for feature selection in the intrusion detection systems

4-2- Evaluation criteria and fitness function

Some criteria, such as accuracy rate, attack detection rate, and false positive rate, can be used for the evaluation of an intrusion detection system [12].

Some of the mentioned criteria are applied in this paper to compare the combinatory and previous methods for the problem of feature selection in intrusion detection systems. Each evaluation criterion is computed by the confusion table in Table 2.

Table 2. Confusion table

		Estimated event type	
		Attack	**Normal**
Real event type	**Normal**	False Positive (FP)	True Negative (TN)
	Attack	True Positive (TP)	False Negative (FN)

The rate of network events that are rightly classified is named the accuracy rate, which should be as high as possible. Equation (6) shows the procedure of computing the accuracy rate.

$$Accuracy\ Rate = \frac{TP + TN}{TN + FP + FN + TP} \quad (6)$$

The rate of network events accurately classified as attacks is named the attack detection rate. As a third measure, the normal network events rate inaccurately classified as attacks is called the false positive rate. Equations (7) and (8) indicate the procedure of computing the DR and FPR, respectively.

$$Detection\ Rate = \frac{TP}{TP + FN} \quad (7)$$

$$False\ Positive\ Rate = \frac{FP}{TP + FN} \quad (8)$$

The fitness function to evaluate the feature subsets obtained by the compared optimization algorithms is created by calculating the accuracy rate. Diverse classifiers is employed in intrusion detection systems, such as neural networks[13]. But,

because the concentration of this paper is over the feature selection process, the naive Bayesian network, created by the selected features, is used as a simple classifier.

4-3- Solution representation of the combinatory algorithm

Search elements of the implemented methods in this paper contain 42 numbers representing the importance of 42 features. The higher numbers in the solution array indicate the selected features of the algorithm. During the iterations of the algorithm, the numbers change in the search space of the feature selection problem. After the last iteration, the S higher numbers related to S important features are selected to build the classifier that will work on the test set records. Equation (9) presents a sample solution array containing ten floating point numbers for ten features. For S=4 selected ones to build a classifier, features 2, 7, 9, and 10 indicate the four higher values.

A sample solution array = [10.3 25 11.2 8 13 1.7 32.4 7.9 45.7 42] (9)

The proposed hybrid algorithm is compared with the genetic algorithm [5], particle swarm optimization [14], gray wolf optimization [15], and the crow search algorithm [9] to solve the feature selection problem in intrusion detection systems. Another important criterion for solving the feature selection problem in intrusion detection systems is the selected features count. It is important to note that, in addition to increasing the processing load in the classification operation, increasing the number of selected features also reduces the accuracy in some cases. For unsophistication, the number of features in the experiments of this paper is determined to be 5, 10, 15, and 20. However the best way to consider all criteria to solve this problem is to use a multi-objective algorithm with an objective function for the number of selected features. Figure and Table 3 compare the accuracy rate obtained from different optimization algorithms for the UNSW-NB15 dataset.

The results in Figure and Table 3 show that the proposed combinatory algorithm has a better accuracy rate for 10 and 15 selected features than the other four compared algorithms. But for 5 and 20 selected features, the accuracy rate of the combined algorithm is lower than the gray wolf and crow search algorithms. Figure and Table 4 show the intrusion detection rate values obtained from the compared algorithms for the UNSW-NB15 data set.

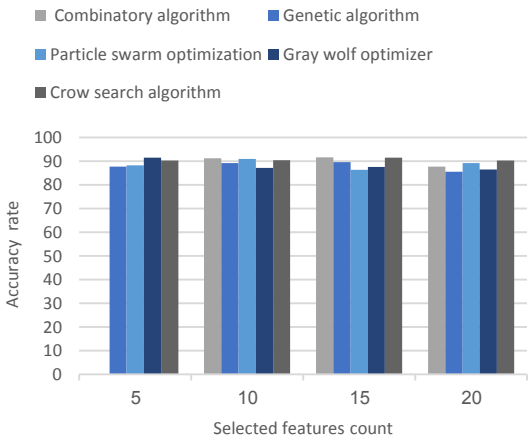

Fig 3. The accuracy rate of the compared algorithms for different selected features count

Table 3. The accuracy rate of the compared algorithms for different selected features count

Algorithm \ Selected Features Count	**5**	**10**	**15**	**20**
Combinatory algorithm	87.96	**91.18**	**91.57**	87.69
Genetic algorithm	87.74	89.14	89.6	85.47
Particle swarm optimization	88.21	90.94	86.38	89.15
Gray wolf optimizer	**91.43**	87.18	87.49	86.43
Crow search algorithm	90.3	90.36	91.51	**90.24**

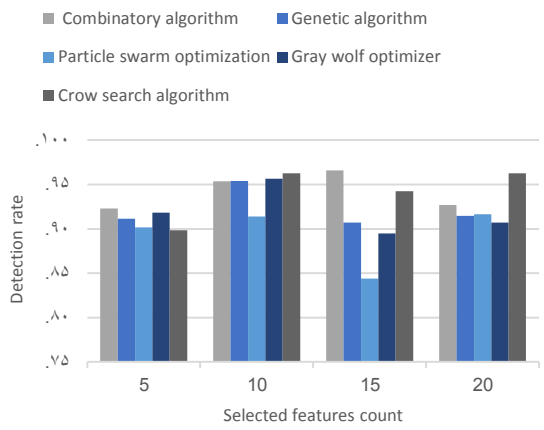

Fig 4. The detection rate of the compared algorithm for different selected features count

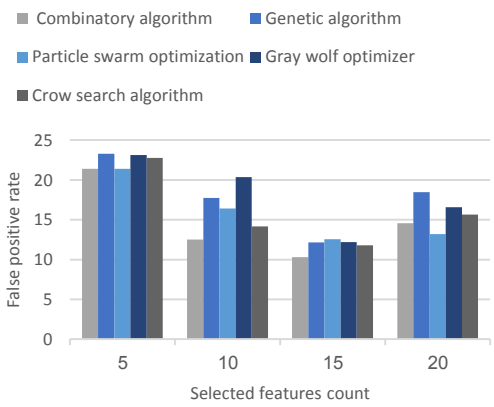

Fig 5. The false positive rate of the compared algorithm for different selected features count

Table 4. The detection rate of the compared algorithm for different selected features count

Selected Features Count / Algorithm	**5**	**10**	**15**	**20**
Combinatory algorithm	**92.29**	**95.35**	**96.59**	92.68
Genetic algorithm	91.14	95.14	90.7	91.45
Particle swarm optimization	90.15	91.38	84.39	91.66
Gray wolf optimizer	91.82	95.64	89.47	90.7
Crow search algorithm	89.85	96.27	94.24	**96.25**

Table 5. The false positive rate of the compared algorithm for different selected features count

Selected Features Count / Algorithm	**5**	**10**	**15**	**20**
Combinatory algorithm	**21.4**	**12.5**	**10.3**	14.58
Genetic algorithm	23.28	17.74	12.17	18.45
Particle swarm optimization	21.41	16.41	12.55	**13.21**
Gray wolf optimizer	23.11	20.36	12.2	16.57
Crow search algorithm	22.78	14.17	11.77	15.67

Figure and Table 4 show that the hybrid algorithm performs better than other algorithms in terms of intrusion detection rate for the number of 5, 10 and 15 features. For the feature number of 20, the intrusion detection rate value for the combinatory algorithm is lower than the crow search algorithm and higher than the others. Figure and Table 5 also show the values of the false positive rate resulting from the compared algorithms for the UNSW-NB15 data set.

As mentioned earlier, the lower false positive rate indicates the higher efficiency of the algorithms for feature selection. Figure and Table 5 also show that the proposed algorithm has a lower false positive rate than other algorithms for the number of 5, 10 and 15 selected features. For the number of features of 20, the particle swarm algorithm has a lower false positive rate.

The results of Figures and Tables 3, 4, and 5 show that 15 is a reasonable number for the number of features selected to create the intrusion detection system classifier. According to the obtained results, the hybrid algorithm can find acceptable answers in most cases for the feature selection problem in intrusion detection systems.

Conclusion

The conducted experiments show that the proposed algorithm provides competitive performance compared to other optimization algorithms for solving the feature selection problem in intrusion detection systems. The reason for a higher functionality of the proposed algorithm is the diversity of answers due to the existence of different movement operators. Therefore, the combinatory algorithm can explore the problem search space well and create diverse answers. On the other hand, in the last iterations of the combinatory algorithm, the focus is on the answers obtained in the exploration phase, and the exploitation phase of the combinatory algorithm finds answers close to the optimal. In future works related to this paper, converting of the combinatory algorithm to a multi-objective one, the parallel implementation of algorithm parts on subpopulations, and the use of neural network classifiers to solve the feature selection problem can be considered.

References

[1] A. S. Eesa, Z. Orman, and A. M. A. Brifcani, "A novel feature-selection approach based on the cuttlefish optimization algorithm for intrusion detection systems," Expert Syst Appl, vol. 42, no. 5, pp. 2670–2679, Apr. 2015, doi: 10.1016/j.eswa.2014.11.009.

[2] 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.

[3] T. Khorram and N. A. Baykan, "Feature selection in network intrusion detection using metaheuristic algorithms," International Journal Of Advance Research, Ideas and Innovations in Technolog, vol. 4, no. 4, pp. 704–710, 2018.

[4] M. H. Aghdam and P. Kabiri, "Feature Selection for Intrusion Detection System Using Ant Colony Optimization," 2016. Accessed: Jun. 06, 2022. [Online]. Available: http://ijns.jalaxy.com.tw/contents/ijns-v18-n3/ijns-2016-v18-n3-p420-432.pdf

[5] Z. Halim et al., "An effective genetic algorithm-based feature selection method for intrusion detection systems," Comput Secur, vol. 110, p. 102448, Nov. 2021, doi: 10.1016/j.cose.2021.102448.

[6] H. Alazzam, A. Sharieh, and K. E. Sabri, "A feature selection algorithm for intrusion detection system based on Pigeon Inspired Optimizer," Expert Syst Appl, vol. 148, p. 113249, Jun. 2020, doi: 10.1016/j.eswa.2020.113249.

[7] T. S. Naseri and F. S. Gharehchopogh, "A Feature Selection Based on the Farmland Fertility Algorithm for Improved Intrusion Detection Systems," Journal of Network and Systems Management, vol. 30, no. 3, pp. 1–27, Jul. 2022, doi: 10.1007/s10922-022-09653-9.

[8] S. Mirjalili, S. M. Mirjalili, and A. Lewis, "Grey Wolf Optimizer," Advances in Engineering Software, vol. 69, pp. 46–61, Mar. 2014, doi: 10.1016/j.advengsoft.2013.12.007.

[9] A. Askarzadeh, "A novel metaheuristic method for solving constrained engineering optimization problems: Crow search algorithm," Comput Struct, vol. 169, pp. 1–12, Jun. 2016, doi: 10.1016/j.compstruc.2016.03.001.

[10] S. Mukherjee and N. Sharma, "Intrusion Detection using Naive Bayes Classifier with Feature Reduction," Procedia Technology, vol. 4, pp. 119–128, Jan. 2012, doi: 10.1016/j.protcy.2012.05.017.

[11] N. Moustafa and J. Slay, "UNSW-NB15: A comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set)," in 2015 Military Communications and Information Systems Conference, MilCIS 2015 - Proceedings, Institute of Electrical and Electronics Engineers Inc., Dec. 2015. doi: 10.1109/MilCIS.2015.7348942.

[12] H. J. Liao, C. H. Richard Lin, Y. C. Lin, and K. Y. Tung, "Intrusion detection system: A comprehensive review," Journal of Network and Computer Applications, vol. 36, no. 1. Academic Press, pp. 16–24, Jan. 01, 2013. doi: 10.1016/j.jnca.2012.09.004.

[13] A. Shenfield, D. Day, and A. Ayesh, "Intelligent intrusion detection systems using artificial neural networks," ICT Express, vol. 4, no. 2, pp. 95–99, Jun. 2018, doi: 10.1016/j.icte.2018.04.003.

[14] A. J. Malik, W. Shahzad, and F. A. Khan, "Network intrusion detection using hybrid binary PSO and random forests algorithm," Security and Communication Networks, vol. 8, no. 16, pp. 2646–2660, Nov. 2015, doi: 10.1002/sec.508.

[15] Q. M. Alzubi, M. Anbar, Z. N. M. Alqattan, M. A. Al-Betar, and R. Abdullah, "Intrusion detection system based on a modified binary grey wolf optimisation," Neural Comput Appl, vol. 32, no. 10, pp. 6125–6137, May 2020, doi: 10.1007/s00521-019-04103-1.