ISSN 2723-6471

Performance Comparison of Whale and Harris Hawks Optimizers with Network Intrusion Prevention Systems

Mosleh M. Abualhaj[1,*], Sumaya N. Al-Khatib[2], Mohammad A. Alsharaiah[3], Mohammad O. Hiari[4]

[1]*Department of Networks and Cybersecurity, Al-Ahliyya Amman University, Amman 19111, Jordan*

[2,3,4] *Department of Data Science and Artificial Intelligence, Al-Ahliyya Amman University, Amman 19111, Jordan*

(Received: July 12, 2024; Revised: July 25, 2024; Accepted: August 24, 2024; Available online: October 15, 2024)

Abstract

Digital technology has permeated every aspect of our daily lives. Processing and evaluating information are highly demanding in all fields, including cybersecurity. Cybersecurity engineers widely use the Network Intrusion Prevention System (NIPS) to safeguard against cyberattacks. To avoid cyberattacks, the NIPS must deal with a large amount of data, which degrades its performance. This paper uses the whale optimization algorithm (WOA) and the Harris Hawks optimization method (HHO) to diminish the large amount of data that the NIPS needs to deal with. Subsequently, the Gradient Boosting Machine (GBM) is employed to determine the accuracy achieved when employing WOA and HHO. The GBM classifier is widely regarded as a sophisticated and straightforward classifier in data mining. Regardless of the premise of feature independence, it outperforms all other classification algorithms by delivering excellent performance. When using GBM, the findings indicate that the accuracy achieved with HHO is 89.81%, but the accuracy attained with WOA is 94.3%.

Keywords: Gradient Boosting Machine, Whale Optimizers, Harris Hawks Optimizers, Intrusion Prevention Systems

1. Introduction

The world is gradually transitioning into the digital realm. A paramount concern is the significant volume of cyberattacks targeting the digital world [1]. The Network Intrusion Prevention System (NIPS) is crucial to safeguard the digital world against cyberattacks. The NIPS can perform several tasks to defeat cyberattacks, including identifying and blocking malicious traffic in real-time, performing deep inspection of the data packet for both header and payload, and monitoring and analyzing the log information from all network systems [2], [3].

The NIPS system employs many ways to identify cyberattacks. The two main methods are signature-based and behavior-based. In the signature-based method, the NIPS system examines the data for a recognized pattern or signature indicative of a cyberattack. The database of signatures is derived from previously identified attacks. In the behavior-based method, the NIPS system analyzes any deviations in the data from its usual patterns. The key advantage of the behavior-based method is that it can detect unknown threats (zero-day attacks). The behavior-based method is a critical component of the modern NIPS system. Recently, the NIPS systems have incorporated machine learning (ML) algorithms to effectively address the emerging sophisticated cyberattacks, including those leveraging artificial intelligence (AI) techniques [2], [3], [4].

The part of artificial intelligence known as machine learning is responsible for creating systems that can learn from the now-available data to generate predictions about the data yet to be received. The capability of the NIPS systems to identify cyberattacks is improved when they are armed with machine learning. However, one of the most significant challenges when merging NIPS systems and ML is the substantial volume of data the NIPS must manage. There is a correlation between the size of the data and the capacity to identify cyberattacks [5]. A subset of ML algorithms, known as feature selection algorithms, are designed to manage massive data and improve the capability of NIPS to identify cyberattacks. The process of decreasing the number of variables input into the NIPS systems by utilizing only the essential data and removing noise from the data is referred to as feature selection. There is a significant category of

***Corresponding author: Mosleh M. Abualhaj (m.abualhaj@ammanu.edu.jo)**
DOI: https://doi.org/10.47738/jads.v5i4.323

feature selection algorithms known as metaheuristic algorithms. These algorithms are built based on modelling concepts found in nature [6]. WOA, which stands for the whale optimization algorithm, and HHO, which stands for the Harris Hawks optimization algorithm, are two of the most well-known metaheuristic algorithms. The Gradient Boosting Machine (GBM) classifier will be utilized in this work to compare the achievement of the WOA and HHO systems with that of the NIPS systems [7], [8].

The remainder of this article is organized as follows: Section 2 showcases a selection of NIPS-related literature on the NSLKDD dataset. Section 3 discusses the suggested method, including analyzing and processing the NSLKDD dataset, discussing the WOA and HHO feature selection algorithms, and customizing the GBM classifier. Section 4 shows the comparison results between the WOA and HHO algorithms. Finally, Section 5 concludes the paper and shows direction for future works.

2. Literature Review

Hanafi et al. [9] employed the Sequential Floating Forward Selection strategy to extract 26 features from the entire dataset. The application involved a two-layer classification process. The first layer utilized Genetic Algorithm Detection Generation (GADG) to distinguish between typical and attack instances. The second layer employed classifiers such as DT, J48, NB, RF Tree, and MLP to classify the attacks. This stage categorizes the attack for a particular category using the NSLKDD and 20% KDD datasets. Each classifier exhibited optimal detection performance for a specific attack while demonstrating suboptimal performance for another type. Vinutha et al. [10] investigated various techniques for selecting features using ensemble and single classifiers on the NSLKDD dataset. Their experimentation has shown that AdaBoost, specifically, significantly improved the classification accuracy. Lee et al. [11] demonstrated the significance of feature selection by examining its influence on enhancing the accuracy of NIPS. The whole NSLKDD dataset is processed using an RF binary classifier in the detection model, utilizing all features without implementing feature selection. Next, a sequential floating search chooses the optimal feature that maximizes the detection rate while minimizing false positives. Gaikwad et al. [12] introduced the GA feature selection strategy to identify the most optimal 15 out of 41 features in the NSLKDD dataset. They assessed the accuracy using the Bagging technique in machine learning to create NIPS, with the partial DT rule serving as the underlying classifier. The results showed that the test dataset's accuracy of 78.37% and the 10-fold cross-validation's accuracy of 99.71% were higher than those of other classifiers. In their study, Ingre et al. [13] utilized the NSLKDD dataset and employed feature reduction techniques employing ratio gain and Artificial Neural Networks (ANN). The results show a commendable achievement of 81.2% in binary classification using 29 features and 79.9% in multi-class classification using 41 features. Subba et al. [14] minimized the dimensionality of the NSLKDD dataset using principal component analysis (PCA) for 17 features. Several classifiers, such as Support Vector Machines (SVM), C4.5, Multi-layer Perceptron (MLP), and Naive Bayes (NB), looked at the 17 features and tested them in both multi-class and binary-class classification. SVM achieved high accuracy in both multi- and binary classifications, outperforming the other classifiers. Pervez et al. [15] presented wrapper feature selection and assessed it using SVM, utilizing the NSLKDD dataset. The results indicated that the SVM attained an accuracy of 91% using three features and 99% accuracy using 41 features on the entire training dataset. However, when tested, the SVM scored 82.37 accuracy with 14 features in binary classification.

3. Method

3.1. NSL KDD Dataset

The achievement of the WOA and HHO algorithms will be compared using the NSL-KDD dataset. The NSL-KDD dataset contains 40 features, excluding the output column. These features are listed in table 1. Besides, the NSL-KDD dataset contains 148,518 samples, divided into 71463 attack samples and 77,055 benign instances. The attack instances distributed over four main types of attack: 53387 samples of Denial of Service (DoS) attack samples, 14077 Probe attack samples, 3880 Remote to Local (R2L) attack instances, and 119 User to Root (U2R) attack instances [16], [17]. The NSL-KDD dataset should be prepared for the machine learning system.

Table 1. NSL-KDD Dataset Features

No	Feature	No	Feature
1	num_file_creations	24	num_outbound_cmds
2	protocol_type	25	num_root
3	dst_host_count	26	num_access_files
4	Flag	27	dst_host_diff_srv_rate
5	service	28	wrong_fragment
11	src_bytes	29	hot
12	srv_rerror_rate	30	is_host_login
13	land	31	num_failed_logins
14	dst_host_same_srv_rate	32	root_shell
15	is_guest_login	33	urgent
16	dst_host_same_src_port_rate	34	dst_host_srv_serror_rate
17	logged_in	35	dst_host_rerror_rate
18	dst_host_srv_diff_host_rate	36	dst_bytes
10	num_compromised	37	count
20	srv_serror_rate	38	srv_count
21	rerror_rate	39	serror_rate
22	diff_srv_rate	40	num_shells
23	srv_diff_host_rate		

First, the label-encoding method converts the textual data into numerical data by replacing each data label with a unique integer number. For example, the Label-encoding method replaces the benign label with 1 and the attack label with 0. Then, the data is scaled into small and close ranges using the Min-max scaling method, in which the large numbers are scaled between 0 and 1[18]. The Min-Max normalization method uses Equation 1. Table 2 show a sample of the NSLKDD dataset before and after preprocessing. Each value in table 2 represents a value of the NSLKDD dataset before and after applying label-encoding and min-max scaling methods.

$$X_{new} = \frac{X - X\_min}{Z\_max - Z\_min} \quad (1)$$

Where Xnew is the new value from the normalized results, X is the old value of the feature, Zmax is the maximum value in the feature, and Zmin is the minimum value in the feature.

Table 2. Samples Before and After Preprocessing

No	Samples Before Preprocessing		Samples After Preprocessing	
	Instances	Output	Instances	Output
1	0, tcp, ftp_data, SF, 491, 0, 0	Benign	0, 0.5, 0.296875, 0.9, 5.48E-06, 0, 0	1
2	0, tcp, private, S0, 0, 0, 0	Attack	0, 0.5, 0.6875, 0.5, 0, 0, 0	0
3	0, tcp, private, S0, 0, 0, 0	Attack	0, 0.5, 0.6875, 0.5, 0, 0, 0	0
4	0, udp, other, SF, 146, 0, 0	Benign	0, 1, 0.625, 0.9, 1.63E-06, 0, 0	1
5	0, tcp, private, REJ, 0, 0, 0	Attack	0, 0.5, 0.6875, 0.1, 0, 0, 0	0

3.2. Feature Selection Using HHO and WOA Optimizers

The key goal of this work is to evaluate the novel HHO and WOA optimizers that improve NIPS system performance. These optimizers both take cues from how animals seek in the wild. The WOA optimizer, for example, takes cues from the way humpback whales hunt, which is to go for small fish in clusters near the ocean's surface. Figure 1 shows that they form unique bubbles around a spiral to surround and seize their prey. Using a bubble net to corner their prey by spiralling toward them, traversing the problem space to identify unexplored zones and boost population diversity, and

 ISSN 2723-6471

seeing the ideal solution as either the target prey or close it inside the search region, the WOA algorithm simulates this behaviour [7], [19].

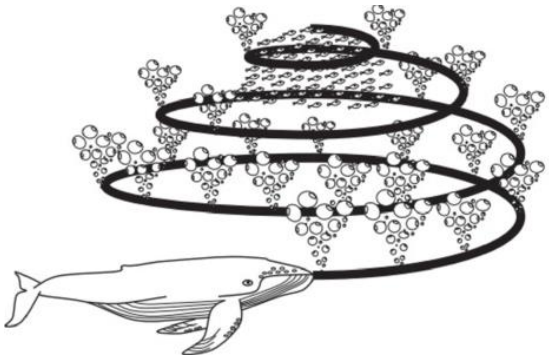

Figure 1. WOA hunting behavior [20]

Conversely, Harris Hawks' natural hunt inspired the HHO optimizer. The HHO approach's primary reasoning is derived from Harris Hawks' natural cooperative behavior and pursuit patterns. Figure 2 illustrates HHO hunting behavior. The HHO offers several search patterns based on random switching statements and concentrates on performance. It is a method for gradient-free optimization with multiple energetic and temporally variable stages of exploitation and exploration tendencies [8], [21], [22].

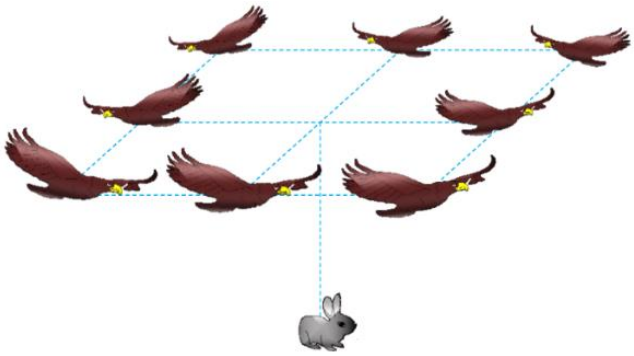

Figure 2. HHO hunting behavior [23]

Despite their simplicity, numerous studies have demonstrated the remarkable efficacy of WOA and HHO in addressing optimization difficulties. Various ways have been developed to handle complex optimization problems in the real world, such as employing WOA and HHO advances, which are known for their user-friendly and easily understandable search algorithms [19], [21], [22].

The HHO and WOA optimizers have been applied to the NSL-KDD dataset to choose the key relevant features that give the best performance to the NIPS systems. The WOA has reduced the NSL-KDD dataset features from 40 to 16. On the other hand, the HHO has reduced the NSL-KDD dataset features from 40 to 13. The features selected by HHO and WOA optimizers are listed in table 4.

Table 4. The selected features by HHO and WOA optimizers

Method	Selected Features
WOA	src_bytes, serror_rate, num_failed_logins, srv_serror_rate, num_outbound_cmds, num_root, is_guest_login, srv_count, Flag, dst_host_same_src_port_rate, srv_diff_host_rate, is_host_login, same_srv_rate, num_access_files, service, and dst_host_rerror_rate
HHO	protocol_type, dst_bytes, hot, dst_host_diff_srv_rate, src_bytes, num_access_files, Flag, urgent, Count, dst_host_srv_count, dst_host_count, diff_srv_rate, and dst_host_same_src_port_rate

3.3. Using GBM for Attack Classification

Even though there are numerous other algorithms in this field, boosting algorithms have earned prevalent popularity in the machine-learning area. The boosting technique adheres to the principle of ensemble learning, wherein it combines numerous elementary models to produce the ultimate output. GBM is widely regarded as one of the most potent boosting algorithms. Gradient boosting fundamentally designs each subsequent model specifically to rectify the errors generated by the preceding models. The ensemble progressively enhances its precision by iteratively reducing the discrepancies and refining the forecasts. The GBM offers several advantages. Firstly, it demonstrates versatility by performing effectively in both classification and regression tasks, making it a flexible algorithm. Secondly, when combined with decision trees, GBM can handle missing values in the data by utilizing surrogate splits during the tree construction process. Lastly, GBM is robust against outliers due to its ensemble approach, which mitigates the impact of individual data points. GBM classifier uses several parameters that impact its performance. Table 5 summarizes the key parameters of the GBM classifier [24], [25].

Table 5. Key parameters of GBM classifier

Method	Selected Features	Assigned Value
n_estimators	Determines the number of boosting stages (or trees) to be used in the ensemble.	100
learning_rate	Controls the contribution of each tree in the ensemble.	0.1
max_depth	Maximum depth of the individual decision trees.	2
min_samples_split	The minimal number of samples needed to divide an internal node.	2
min_samples_leaf	The minimal number of samples necessary to form a leaf node.	1
subsample	The proportion of samples utilized for training the individual base learners.	1.0
max_features	The quantity of features to take into account when searching for the optimal division.	None

4. Result and Discussion

The comparison of WOA and HHO was conducted on a PC with Intel Core i9 13900K 3.0 GHz (24 Core, 32MB L2 Cache, 32 total threads, up to 5.8 GHz), 24 GB RAM, 1TB SSD, and Ubuntu 21.10 O.S. The GBM, HHO, and WOA were deployed using Python. In addition, the K-Fold cross-validation mechanism has been utilized to divide the NSL-KDD dataset into five groups to validate the proposed model. Several libraries from Python 3.12 were used including 'sklearn.preprocessing', mealpy.swarm_based.HHO, 'GradientBoostingClassifier', 'mealpy.swarm_based.WOA, train_test_split', 'numpy', and 'pandas' [26], [27].

The comparison of WOA and HHO optimizers is achieved using Accuracy, Precision, and Recall metrics. These metrics are derived from True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN). The accuracy determines the true attack identifications (TP and TN) ratio in all samples. Equation 2 is used to find the accuracy. Equation 3 is used to find the Precision. Precision determines the correct positive attack identification ratio to the number of positive attack identifications. The recall calculates the ratio of correctly identified positive attacks to actually identified positives. Equation 4 is used to find the Recall [26], [27].

$$\text{Accuracy}=(TP+TN)/(TP+TN+FP+FN) \quad (2)$$

$$\text{Precision}=TP/(TP+FP) \quad (3)$$

$$\text{Recall}=TP/(TP+FN) \quad (4)$$

Regarding Accuracy, figure 3 compares the WOA and HHO optimizers using the GBM classifier. The two optimizers (WOA and HHO) have accurately detected attacks using NIPS systems. The results show that HHO achieved an Accuracy of 89.81%, while WOA achieved an Accuracy of 94.3%. The WOA optimizer achieved a considerable Accuracy improvement of 4.49% compared to the HHO optimizer. For this reason, when utilizing NIPS systems to locate the attack, WOA was found to be more accurate than HHO when the GBM classifier and the NSL-KDD dataset were considered.

Regarding Precision, figure 4 compares the WOA and HHO optimizers using the GBM classifier. The two optimizers (WOA and HHO) have good Precision in detecting attacks using NIPS systems. The results show that HHO achieved a Precision of 89.81%, while WOA achieved a Precision of 94.3%. The WOA optimizer achieved a considerable Precision improvement of 4.49% compared to the HHO optimizer. For this reason, when utilizing NIPS systems to

ISSN 2723-6471

locate the attack, WOA was found to be more accurate than HHO when the GBM classifier and the NSL-KDD dataset were considered.

Regarding Recall, figure 5 compares the WOA and HHO optimizers using the GBM classifier. The two optimizers (WOA and HHO) have good Recall detecting attacks using NIPS systems. The results show that HHO achieved a Recall of 89.81%, while WOA achieved 94.3%. The WOA optimizer achieved a considerable Recall improvement of 4.49% compared to the HHO optimizer. For this reason, when utilizing NIPS systems to locate the attack, WOA was found to be more accurate than HHO when the GBM classifier and the NSL-KDD dataset were considered.

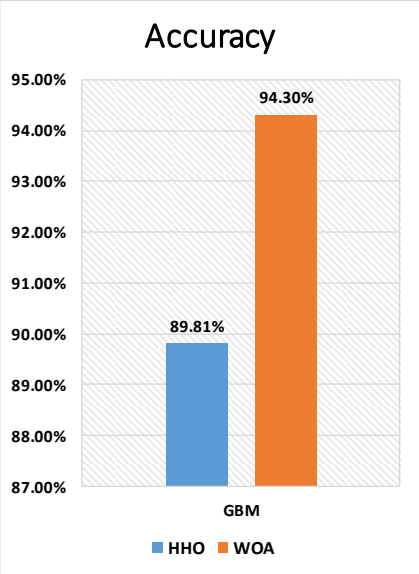

Figure 3. Accuracy of the WOA and HHO algorithms

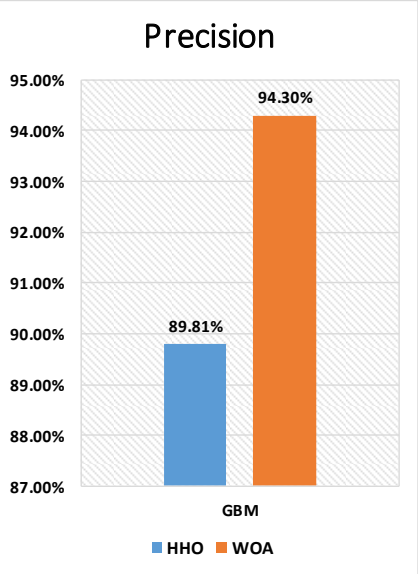

Figure 4. Precision of the WOA and HHO algorithms

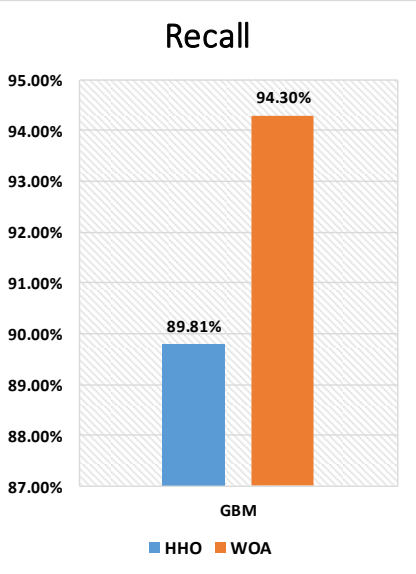

Figure 5. Recall of the WOA and HHO algorithms

In summary, the two algorithms (WOA and HHO) have performed well with the NIPS system. However, the WOA algorithm has outperformed the HHO algorithm. This is because the WOA algorithm provides a balance between exploration and exploitation. Therefore, when searching the feature space, the WOA algorithm is more capable of selecting the most important features without redundant feature selection. On the other hand, though the HHO algorithm can rapidly converge, it will mostly converge to suboptimal feature subsets, degrading its performance. As a result, the WOA algorithm selects more general features, leading to better NIPS system performance than HHO's.

5. Conclusion

This study introduces two feature selection methods: the WOA and the HHO algorithms. Feature selection methods are crucial, particularly for a NIPS developed for large-scale networks with high volume and velocity. This study compared the WOA and HHO selection methods using the well-known NSLKDD99 dataset. The chosen characteristics were thoroughly assessed and contrasted using a GBM classifier. The experimental results demonstrated that the feature set selected by the WOA optimizer outperformed the feature set selected by the HHO optimizer, indicating its potential and suitability for large-scale network NIPS systems. The results demonstrate that the accuracy obtained using HHO is 89.81%, but the accuracy gained with WOA is 94.3%. Accordingly, adopting the WOA algorithm for feature selection in the NIPS systems will improve the performance, particularly in complex datasets with intricate feature interactions. Future research will investigate the performance of the WOA algorithm with other attack datasets. In addition, different feature selection algorithms will be studied using the NIPS systems.

6. Declaration

6.1. Author Contributions

Conceptualization: M.M.A., S.N.A., M.A.A., and M.O.H.; Methodology: M.O.H.; Software: M.M.A.; Validation: M.M.A., S.N.A., M.A.A., and M.O.H.; Formal Analysis: M.M.A., S.N.A., M.A.A., and M.O.H.; Investigation: M.M.A.; Resources: M.O.H.; Data Curation: S.N.A.; Writing Original Draft Preparation: M.M.A., S.N.A., M.A.A., and M.O.H.; Writing Review and Editing: S.N.A., M.M.A., M.A.A., and M.O.H.; Visualization: M.M.A.; All authors have read and agreed to the published version of the manuscript.

6.2. Data Availability Statement

The data presented in this study are available on request from the corresponding author.

6.3. Funding

The authors received no financial support for the research, authorship, and/or publication of this article.

6.4. Institutional Review Board Statement

Not applicable.

6.5. Informed Consent Statement

Not applicable.

6.6. Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] M. Bang and H. Saraswat, "Building an effective and efficient continuous web application security program," *2016 International Conference On Cyber Situational Awareness, Data Analytics and Assessment (CyberSA), London, UK*, vol. 2016, no. Jun., pp. 1-4, 2016, doi: 10.1109/CyberSA.2016.7503287.

[2] M. M. Abualhaj, A. A. Abu-Shareha, Q. Y. Shambour, A. Alsaaidah, S. N. Al-Khatib, and M. Anbar, "Customized K-nearest neighbors' algorithm for malware detection," *Int. J. Data Netw. Sci.*, vol. 8, no. 1, pp. 431–438, 2024. DOI: 10.5267/j.ijdns.2023.9.012.

[3] M. M. Abualhaj, Ahmad Adel Abu-Shareha, M. O. Hiari, Yousef Alrabanah, Mahran Al-Zyoud, and M. A. Alsharaiah, "A Paradigm for DoS Attack Disclosure using Machine Learning Techniques," *International Journal of Advanced Computer Science and Applications*, vol. 13, no. 3, pp. 192-200, Jan. 2022.

[4] H. Alazzam, A. Sharieh, and K. E. Sabri, "A feature selection algorithm for intrusion detection system based on Pigeon Inspired Optimizer," *Expert Systems with Applications*, vol. 148, no. 1, pp 1-14, Jun. 2020, doi: 10.1016/j.eswa.2020.113249.

[5] H. Mendes, S. E. Quincozes, and V. E. Quincozes, "A Web User Interface Tool for Metaheuristics-Based Feature Selection Assessment for IDSs," *in 6th Cyber Security in Networking Conference (CSNet), Rio de Janeiro, Brazil*, vol. 2022, no. Oct., pp. 24-26, 2022, IEEE, 2022. DOI: 10.1109/csnet56116.2022.9955616.

[6] M. Kolhar, F. Al-Turjman, A. Alameen, and M. M. Abualhaj, "A three layered decentralized IoT biometric architecture for city lockdown during COVID-19 outbreak," *IEEE Access*, vol. 8, pp. no. 1, 163608-163617, 2020. DOI: 10.1109/ACCESS.2020.3021983.

[7] M. Alazab, R. Abu Khurma, P. A. Castillo, B. Abu-Salih, A. Martín, and D. Camacho, "An effective networks intrusion detection approach based on hybrid Harris Hawks and multi-layer perceptron," *Egyptian Informatics Journal*, vol. 25, no. 1, pp. 100423-100432, Feb. 2024, doi:10.1016/j.eij.2023.100423.

[8] S. Mirjalili and A. Lewis, "The Whale Optimization Algorithm," *Advances in Engineering Software*, vol. 95, no. 1, pp. 51-67, May 2016. DOI: 10.1016/j.advengsoft.2016.01.008.

[9] A. S. A. Aziz, S. E.-O. Hanafi, and A. E. Hassanien, "Comparison of classification techniques applied for network intrusion detection and classification," *Journal of Applied Logic*, vol. 24, no. 1, pp. 109–118, Nov. 2017.

[10] H. P. Vinutha and B. Poornima, "An Ensemble Classifier Approach on Different Feature Selection Methods for Intrusion Detection," *Advances in intelligent systems and computing*, vol 672, no. 1, pp. 442–451, Jan 2018, doi: 10.1007/978-981-10-7512-4_44.

[11] J. Lee, D. Park, and C. Lee, "Feature Selection Algorithm for Intrusions Detection System using Sequential Forward Search and Random Forest Classifier," *KSII Transactions on Internet and Information Systems*, vol. 11, no. 10, pp. 5132-5148, Oct. 2017. DOI: 10.3837/tiis.2017.10.024.

[12] D. Gaikwad and R. C. Thool, "Intrusion detection system using bagging with partial decision tree base classifier," *Procedia Computer Science*, vol. 49, no. 1, pp. 92–98, 2015. doi: 10.1016/j.procs.2015.04.227.

[13] B. Ingre and A. Yadav, "Performance analysis of NSL-KDD dataset using ANN," *2015 International Conference on Signal Processing and Communication Engineering Systems, Guntur, India*, vol. 2015, no. Mar., pp. 92-96, 2015, doi: 10.1109/SPACES.2015.7058223.

[14] B. Subba, S. Biswas and S. Karmakar, "Enhancing performance of anomaly based intrusion detection systems through dimensionality reduction using principal component analysis," *2016 IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS), Bangalore, India*, vol. 2016, no. Jun., pp. 1-6, 2016, doi: 10.1109/ANTS.2016.7947776.

[15] M. S. Pervez and D. M. Farid, "Feature Selection and Intrusion Classification in NSL-KDD Cup 99 Dataset Employing SVMs," *in Proceedings of the 8th International Conference on Software, Knowledge, Information Management and Applications (SKIMA 2014), Dhaka, Bangladesh*, vol. 2014, no. Dec., pp. 1-6, 2014, doi: 10.1109/SKIMA.2014.7083539.

[16] H. Al-Mimi, N. A. Hamad, M. M. Abualhaj, M. S. Daoud, A. Al-Dahoud, and M. Rasmi, "An Enhanced Intrusion Detection System for Protecting HTTP Services from Attacks," *International Journal of Advances in Soft Computing and Its Applications*, vol. 15, no. 3, pp. 67-84, 2023.

[17] Ü. Çavuşoğlu, "A new hybrid approach for intrusion detection using machine learning methods," *Applied Intelligence*, vol. 49, no. 7, pp. 2735–2761, Jul. 2019.

[18] H. Al-Mimi, N. A. Hamad and M. M. Abualhaj, "A Model for the Disclosure of Probe Attacks Based on the Utilization of Machine Learning Algorithms," *2023 10th International Conference on Electrical and Electronics Engineering (ICEEE), Istanbul, Turkiye*, vol. 2023, no. Dec., pp. 241-247, 2023, doi: 10.1109/ICEEE59925.2023.00051

[19] K. Dev, P. K. R. Maddikunta, T. R. Gadekallu, S. Bhattacharya, P. Hegde and S. Singh, "Energy Optimization for Green Communication in IoT Using Harris Hawks Optimization," *in IEEE Transactions on Green Communications and Networking*, vol. 6, no. 2, pp. 685-694, June 2022, doi: 10.1109/TGCN.2022.3143991.

[20] Z. Yu, X. Shi, J. Zhou, X. Chen, and X. Qiu, "Effective Assessment of Blast-Induced Ground Vibration Using an Optimized Random Forest Model Based on a Harris Hawks Optimization Algorithm," *Applied Sciences*, vol. 10, no. 4, pp. 1403-1419, Feb. 2020, doi: 10.3390/app10041403.

[21] C. Li, C. You, Y. Gu and Y. Zhu, "Parameter Identification of the RBF-ARX Model Based on the Hybrid Whale Optimization Algorithm," *in IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 71, no. 5, pp. 2774-2778, May 2024, doi: 10.1109/TCSII.2024.3351848.

[22] M. H. Nadimi-Shahraki, H. Zamani, Z. A. Varzaneh, and S. Mirjalili, "A Systematic Review of the Whale Optimization Algorithm: Theoretical Foundation, Improvements, and Hybridizations," *Archives of Computational Methods in Engineering*, vol. 30, no. 7, pp. 4113-4159, Jul. 2023, doi: 10.1007/s11831-023-09861-3.

[23] L. Diop, S. Samadianfard, A. Bodian, Z. M. Yaseen, M. A. Ghorbani, and H. Salimi, "Annual rainfall forecasting using hybrid artificial intelligence model: integration of multilayer perceptron with whale optimization algorithm," *Water Resources Management*, vol. 34, no. 2, pp. 733-746, 2020, doi: 10.1007/s11269-019-02473-8.

[24] R. Wen and K. Zhang, "Research on Automated Classification Method of Network Attacking Based on Gradient Boosting Decision Tree," *2022 International Conference on Machine Learning and Knowledge Engineering (MLKE), Guilin, China*, vol. 2022, no. Feb., pp. 72-76, 2022, doi: 10.1109/MLKE55170.2022.00019.

[25] Q. Y. Shambour, M. M. A. Alhaj, and M. M. A. Tahrawi, "A hybrid collaborative filtering recommendation algorithm for requirements elicitation," *International Journal of Computer Applications in Technology*, vol. 63, no. 1/2, p. 135, 2020, doi: 10.1504/ijcat.2020.107908.

[26] A. Al Saaidah, "Enhancing malware detection performance: leveraging K-Nearest Neighbors with Firefly Optimization

Algorithm", *Multimed Tools Appl*, vol. 2023, no. 1, pp. 1-12, 2024. doi: 10.1007/s11042-024-18914-5

[27] H. M. Al-Mimi, N. A. Hamad, M. M. Abualhaj, S. N. Al-Khatib, and M. O. Hiari, "Improved Intrusion Detection System to Alleviate Attacks on DNS Service", *Journal of Computer Science*, vol. 19, no. 12, pp. 1549-1560, 2022, doi: 10.3844/jcssp.2023.1549.1560.