United International Journal for Research & Technology
Volume 04, Issue 08, 2023 | Open Access | ISSN: 2582-6832

Enhancement of Random Forest by Utilizing Modified Whale Optimization Algorithm

Ali Ojiram G. Pirzada[1], Renz Michael M. Leandicho[2], Mark Christopher R. Blanco[3], Raymund M. Dioses[4], and Vivien A. Agustin[5]
[1,2]Student, College of Engineering - Pamantasan ng Lungsod ng Maynila
[3,4,5]Professor, College of Engineering - Pamantasan ng Lungsod ng Maynila

***Abstract*—** Machine learning (ML) has various applications, including the ability of software to predict and analyze results more correctly without explicit instructions, identify the best ways to automate tasks, enhance processes, and many other things. The Random Forest (RF) model has been proven to perform well and has applications in many different sectors, but current research suggests that there is still room for improvement. It is the most well-known and often used machine learning technique. There is still room for development with the RF model. In this paper, the researchers provided an optimization algorithm (WOA) to enhance and improve the accuracy of the Random Forest Algorithm on a UNSW-NB15 Intrusion detection dataset. It achieved an accuracy of 97.14% with the hybrid algorithm compared to the traditional algorithm of 94.79%. Furthermore, the recall scores for the proposed algorithm and traditional RF were 95.80% and 92.26%, respectively, while the precision scores for MWOA-RF and traditional RF were equal at 1.000. It indicates that the suggested method performed better at correctly identifying positive cases and had a lower rate of false negatives recognized. Lastly, The F1-Score given by the MWOA-RF is 0.9785 compared to the F1-Score of the traditional RF, which is 0.9597, which signifies that the proposed MWOA-RF performs better for classification and is the better model of the two since its value is closer to 1.

***Keywords*—** Machine Learning, Metaheuristic, Random Forest, Whale Optimization Algorithm, Overfitting, Hyperparameter tuning, Feature Selection, Classification, Confusion Matrix, Precision, Recall.

I. INTRODUCTION

Today, the complexity and interconnectedness of technologies are growing. [1] the majority of devices, machines, and everything we use now produce data due to the digitalization of society. These advancements have already made significant impacts in this era. In fact, [2] states that technology may also be an immense source of the future. This means the role of technology in shaping the future is vast, and one particular area that holds significant influence is the development of machine learning-related technologies. Machine learning induces self-learning in computers without explicit programming.

Nowadays, there are many uses for machine learning (ML), such as software programs that can predict and analyze outcomes more accurately without having to be explicitly instructed to do so or find the most effective solutions to automate jobs, improve procedures, and many more. Several of the best learning algorithms are random forests [3]. Moreover, the Random Forest Algorithm is the most popular and commonly used machine learning algorithm [4].

Although the Random Forest (RF) model has been shown to perform well and has applications in many different fields, ongoing research suggests that there is still room for advancement. The RF model still has space for improvement, despite its high performance and widespread application. [5] [6] iterates that the biggest drawback of random forest is that it can be too sluggish and inefficient for real-time forecasts when there are many trees. This might be incredibly challenging if the model is used in fast-responding applications, such as real-time data processing or online systems.

Additionally, [7] states that models created using a large number of trees encode more complexity than those created using a small number of trees. This means that a possibility of sizeable random forest models may overfit the dataset they were trained on. Hence, classification algorithms such as Random Forest still suffer from overfitting. As mentioned by [8]. the over-fitting problem is addressed using Random Forest classifiers by modifying its hyper-parameters and analyzing the results. Poor hyperparameter tuning might lead to an overfitting phenomenon, and that affects the performance of the model, such as its classification and feature selection, thus leading to problems for optimization; therefore, the hyper-parameters of the algorithm need to be tweaked to prevent overfitting in Random Forest [9].

Some developments may help increase the RF model's general efficacy and applicability across various areas. Current approaches for enhancing and solving the problems of traditional algorithms are using optimization algorithms; optimization requires a global iterative search algorithm. This approach has been seen in the study of [10]. While other approaches are being made to enhance traditional algorithms, this study proposes using the Whale Optimization Algorithm (WOA), which is presently used to optimize problems and is being developed to achieve a significantly more accurate result.

It is also integrated with several algorithms to assist other algorithms in generating even improved performances. Since overfitting, optimization, and feature selection are the three particular challenges with the Random Forest technique addressed in this study, this paper suggests using the Random Forest method combined with the Modified Whale Optimization Algorithm (WOA) to address these issues.

In order to address these issues, the study suggests an approach that combines the strengths of the Random Forest algorithm and the Modified Whale Optimization Algorithm (MWOA).

By utilizing the MWOA's optimization capabilities, the Random Forest model may achieve better generalization, improved parameter settings, and enhanced feature selection, ultimately producing more accurate and effective predictions.

1.1 Statement of the Problem

The following problems were found in the traditional Random Forest:

a) Large random forest models may overfit the training dataset, yet with fewer trees in the random forest when the model is not complex enough. It could not function efficiently, raising the optimization problem.
b) Random forest model is prone to improper tuning in terms of its hyperparameters that tend to underperform for classification and prediction tasks.
c) The random forest feature selection does not show the actual relevance of the predictors for large data sets that may overfit.

1.2 Objectives of the Study

a) The intention of reducing overfitting issues and enhancing models by using modified WOA and Random Forest.
b) To modify the Whale Optimization Algorithm for hyperparameter tuning of Random Forest to offer potential improvements to RF and help increase the model's generalization performance and accuracy using a modified approach by determining the ideal hyperparameter settings.
c) To make precise predictions on unobserved data and modify the feature selection by incorporating the modified Whale Optimization Algorithm

II. RELATED WORKS

2.1 Random Forest Algorithm

Random forests are machine learning models that combine the results from a series of regression decision trees, or "forests," to create output predictions [11]. This works by assembling a group of decision trees into a "forest" during the training stage. Each decision tree is independently trained on a random portion of the training data, and a random subset of characteristics is considered at each split point.

The main practical advantage of utilizing Random Forest is that it automatically corrects for decision trees' ability to overfit their training set. In most cases, Random Forest's accuracy can be maintained even when some data is absent [12]. Not only that, but the advantage of the Random Forest method is its capacity to analyze big datasets with a higher spatiality [4].

According to [3] ensemble learning methods like random forests are well-suited for medium-sized to big datasets.

A random subset of features is then examined to find the best one, and due to the enormous variation created by this, the model is generally better. However, according to [13], the number of features is the single factor influencing the model complexity in a random forest.

As a result, if a particular feature has a high correlation with the dependent variable, it can be divided into numerous bins, and each bin will then receive its own tree, therefore, causing it to overfit because when there is a highly significant association between a dependent variable and a feature, overfitting occurs. This means that irrelevant traits could still cause harm to Random Forest [14].

Therefore, according to [15], it can be deduced that Random Forest still needs to have its performance optimized for using datasets. Although Random Forest is a strong and flexible algorithm, its performance can be enhanced for particular datasets and use cases to increase accuracy, interpretability, and overall effectiveness. There are recent studies of enhancing Random Forest Algorithms with the use of other techniques.

In the study by [16], they improved Random Forest by merging it with three other Feature Selection approaches (Chi-Squared, Random Forest (RFI), and Linear Correlation (LC) Filter) and a resampling method (Random OverSampling (ROR)). Results show that the suggested method outperforms RFA for both datasets, with improvements in R2 values of 69.50%, 65.57%, and 69.40% for QUES and 31.90%, 44.94%, and 51.81% for UIMS, respectively, using chi-squared, RF, and linear correlation filter approaches. However, the study can be further developed by including additional datasets, different ML, FS, and resampling approaches, and various prediction accuracy metrics. Other studies use different algorithms, such as the study by [17], which uses reinforcement learning to improve the performance of the Random Forest algorithm that, according to the data, produced the suggested strategy, performs better with lower error measures and enhanced accuracy of 92.2%.

In addition, the study by [18] uses Principal Component Analysis (PCA) and Random Forest together to assess the condition of hydrogen fuel cells, and the results of the study demonstrate that the mentioned algorithm performs better than alternative techniques. When a model is being tweaked, machine learning turns from science to trial and error since the ideal hyperparameters are frequently difficult to anticipate in advance. Herein lies the significance of hypertuning and optimization of random forests [19]. In fact, according to [20], the Bayesian optimization technique is then used to optimize the Random Decision Forest classifier model to find the best hyper-tuned parameters. The critical phase in the development process of a machine-learning model is hyperparameter tuning, which can increase the model's accuracy [21].

Moreover, [30] enhanced the functionality of the Random Forest (RF) model using six metaheuristic optimization techniques, and plenty more studies are now using metaheuristic algorithms to improve traditional algorithms such as Random Forests. Studies like those conducted by [22] utilize better Artificial Fish Swarm in conjunction with Random Forest improvement to forecast the medial knee contact force, producing a Pearson correlation coefficient of 0.970. This shows that the suggested model can correctly determine the causal connection between the inputs and results. Another hybrid algorithm that combines classic and metaheuristic techniques, like [23] application-specific clustering in wireless sensor networks, employs a combination of the fuzzy firefly algorithm and random forest. These are some examples of hybrid algorithms that improve conventional algorithms and might be useful for enhancing performance.

2.2 Whale Optimization Algorithm

The process of iteratively increasing a machine learning model's accuracy and reducing its level of error is known as machine learning optimization [24]. It trains the model iteratively through optimization, which yields an assessment of the maximum and minimum functions [25]. According to [26], the Gradient Descent Algorithm and its derivatives, the Stochastic Gradient Descent and the MiniBatch Gradient Descent, are widely used iterative approaches for solving optimization issues in machine learning. One innovative method for handling optimization issues is the Whale Optimization Algorithm (WOA).

The whale optimization algorithm is a swarm-based intelligence optimization algorithm with a strong global search capability thanks to its distinctive search methodology [27]. According to [28], it has been demonstrated that this algorithm performs as well as or even performs better than some of the existing algorithmic strategies. [29] state that the Whale Optimization Algorithm (WOA) has produced better outcomes compared to other algorithms. The outcomes demonstrate how effective and reliable the WOA is. However, like all other algorithms, there are also drawbacks when it comes to WOA. The basic whale optimization method has the disadvantages of search stagnation, simple local optimum entry, slow convergence, and low calculation accuracy. According to [30], to overcome the shortcomings of the original WOA, including delayed convergence, stagnation at local minima, and poor stability, a study was developed as modified whale optimization based on multiple strategies, known as MSWOA. This statement was supported by [31].

An improved whale optimization algorithm (IWOA) was also presented in response to the limitations of the whale optimization algorithm (WOA), including its slow convergence speed, poor accuracy, and propensity for local optimum [32]. To overcome the shortcomings, the study of [33] suggested using the new whale optimization algorithm (HMNWOA). The original method's ability to do global searches is carried over into the suggested approach, improving the population's quality, exploitation potential, and convergence speed. Therefore, whale optimization algorithms are now being used to optimize and enhance many studies and other algorithms. [34] states that various types of research have been conducted on WOA.

According to [35], in 5 areas and 17 subfields of diverse engineering domains, WOA-based methodologies are used. There has been 61% effort on WOA technique modification, 27% work on hybridization, and 12% work on multi-objective variations. Another study from [36] proposed a hybrid SA-WOA algorithm that simulated the annealing (SA) algorithm, which was added to the process by examining the most promising regions found by basic WOA in order to improve the utilization of WOA. The study's experimental findings support the effectiveness of the suggested strategies in increasing classification accuracy when compared to other wrapper-based algorithms.

III. METHODOLOGY

The proposed enhancement for the existing algorithm is illustrated in Fig 3.1

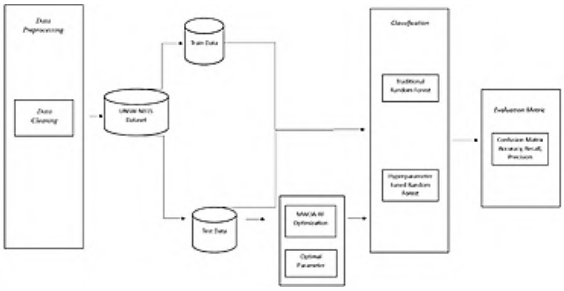

Fig 3.1. *Overall Diagram of the Methodology*

Figure 3.1 above shows the summary of the entire methodology of this study. First, the process begins with data preprocessing, where techniques such as label encoding, and data cleaning are performed. Second, the dataset was divided into training and test data. The proposed MWOA-RF applied the hyperparameter tuning and optimization and, lastly, the classification and evaluation of the existing and proposed algorithm.

3.1 Data Preprocessing

One of the methodologies used is data preprocessing, an essential step in the pipeline for data analysis and machine learning. According to [37], preprocessing entails putting raw data into a form the model can understand.

Table 3.1 *Number of records in training and testing subsets for each class*

Classes	Training Subset	Testing Subset
Normal	56,000	37,000
Analysis	2,000	677
Backdoor	1,746	583
DoS	12,264	4,089
Exploits	33,393	11,132
Fuzzers	18,184	6,062
Generic	40,000	18,871
Reconnaissance	10,491	3,496
Shellcode	1,133	378
Worms	130	44
Total Number of Records	**175,341**	**82,332**

The data used in the study is a dataset for network intrusions named UNSW-NB15. It has nine attacks, including worms, backdoors, DoS attacks, and fuzzers.

Raw network packets are included in the collection. 175,341 records make up the training set, while 82,332 records from the attack and normal types make up the testing set out of the initial 2,218,761 records. The researchers used the UNSW-NB15 training and testing subsets that [38] published and made.

The researchers also performed Label Encoding, a technique for numerically representing categorical variables. It gives each category in the feature/column a distinct integer value. The algorithm uses this to identify the categorical features in the training set. Encoding of

categorical features for training and testing set is applied using Label Encoding.

3.2 Data Cleaning

Data was cleaned by locating and fixing problems, including missing values and noisy data. Rows within the dataset containing redundant values and data are also removed from the data frame. his eliminates the data's inconsistencies and impurities, making it more efficient when performing analysis within the dataset [39].

3.3 Dataset

The dataset used was divided into two parts, the training and testing dataset. The training dataset was used for the Traditional Random Forest and for tuning hyperparameters through the Modified Whale Optimization Algorithm (MWOA).

It was tested using a validation dataset with the optimal parameters. A testing dataset provides a neutral assessment of a finished model. This evaluates the performance and progress of algorithms' training and adjusts or optimizes it for improved results [40].

3.4 Traditional Random Forest Algorithm

The first stage in comparing the accuracy and performance of the traditional Random Forest Algorithm to the experimental approach that this study wants to propose is to run the traditional RF on the UNSW NB-15 dataset, which was split into two partitions. This approach allows for a rigorous assessment of the algorithm's precision and reliability, providing a baseline for comparison to the study's proposed experimental approach.

After preparing and splitting the data, the researchers will perform the usual data preprocessing technique by encoding categorical features on training and test datasets. This process aids in cutting down on overfitting and computing complexity.

The researchers then set the hyperparameters for the Random Forest algorithm's initialization, including the number of trees in the forest (n_estimators), the number of features that can be split into as many categories as possible (max_features), and the reproducibility seed (random_state). By fitting the data to the algorithm, train the Random Forest model with the training set. In this step, decision trees are grown, the best splits are chosen, and an ensemble of trees is constructed. Using the test set, the researchers assessed how well the trained Random Forest model performed. To evaluate the model's performance in producing predictions, compute several evaluation metrics like accuracy, precision, and recall. These measurements aid in determining the model's advantages and disadvantages and, if necessary, serve as a guideline for modifications.

3.5 Modified Whale Optimization Algorithm

The modified whale optimization algorithm used in this study utilizes improved areas. Mainly, they are the population size, cooperative behavior, exploration and exploitation, number of iterations, and modified fitness equation attuned for the Random Forest algorithm. MWOA was first used in the training dataset to find the optimal hyperparameters and later validated using a separate testing dataset.

On the population size multiple whales were used that were different from the usual single whale that is generally used in the Whale Optimization Algorithm. This helped in finding the best solution for the parameters. The variety of whale populations used presented a wider search area that can be explored and much more diverse solutions.

The fitness function is used for evaluating the configuration of the Random Forest based on accuracy. Each whale and the number of trees is used to assess the performance which was used for optimizing to avoid overfitting through hyperparameter tuning.

The features of the dataset and the selected features of the MWOA are defined as the search space for the whale to explore. To optimize the efficiency of finding the hyperparameters, a modified update equation was used. This would balance out the exploration and exploitation phase, as to which it dynamically adjusts as it iterates. MWOA can effectively direct the search process towards promising areas of the search space by including more terms and coefficients. A threshold has been set to the 75th percentile in choosing the best feature values. These features are identified as significant to be chosen.

The initial population of whales and the number of iterations were tested through trial and error. The parameters used in finding the best mtry were 30, 35, 40, 45, 50 iterations, and 15, 20, 25, 30 whales to get the best possible combination. Values exceeding those numbers result in overfitting, and diminishing returns, thus not suited for hyperparameter tuning or initializing

the population of whales and iterations for n_estimators, the values are set for the whale to randomly explore and exploit from a range of 1 to 500 trees. Together with the selected features, the whale would then find the best n_estimators in terms of accuracy. The results of the whale were used and tested on the modules of Random Forest and evaluated using the respective evaluation metric.

3.6 Hyperparameter Tuning of Random Forest

The optimal parameter results produced by MWOA-RF with the best fitness value are applied and utilized in a Random Forest Algorithm. This type of study and combination is related to the works when the introduction of metaheuristics enhances machine learning approaches. [41] states that using metaheuristics has greatly enhanced the performance of machine learning tasks. This will be beneficial in determining whether the parameters developed by the proposed algorithm (a combination of a metaheuristic and ML) may provide improved accuracy and effectiveness.

In applying the parameter results to the Random Forest Algorithm, the initial phase is similar regarding data preprocessing. Data Loading or importing necessary libraries on the IDE along with two datasets that contain the training and testing data is conducted. The loaded datasets are stored as DataFrames using the variables train_df and test_df. As mentioned earlier, the datasets for training and testing are inserted into the variables to perform data loading.

Afterward, data cleaning was conducted using the dropna() method to remove rows from the training to build the classifier and test datasets with missing values.

Adding LabelEncoder is the next procedure, as it encodes categorical features in the training and testing datasets. The code uses the select_dtypes() method to find the categorical columns in each dataset. It then generates a LabelEncoder object and applies it to each categorical feature to convert it to a numerical representation. Machine learning models often require numerical inputs, therefore, this phase is essential. It will be applied when encoding categorical features for training and testing datasets.

The features and labels are divided into both the training and testing datasets. The code shown below is the variables X_train and X_test, which are given the features (X), whereas Y_train and Y_test is given the labels (Y). By excluding the last column, it is essentially excluding the target variable or the labels from the feature set that is why a (‘-1’) indexing is being used to select specific columns on the Dataframe. To train the machine learning model on the input features and assess its performance on the associated target labels, these two processes must be separated.

By partitioning the data, researchers may train the machine learning model on certain portions of the data and assess its effectiveness on previously unseen data. This model is evaluated on how well it can predict labels in a testing dataset after learning from the correlation between features and labels in the training dataset. This may test the model's versatility and capacity to make precise predictions in real-world circumstances by measuring how effectively it performs on unseen data.

When the features and labels of both the training and testing dataset are separated already, a Random Forest classifier will be produced with certain parameters, including the number of estimators, the maximum number of features, and the random state. To build the classifier, the RandomForestClassifier() method is invoked with these arguments. [42] states that Random forest hyperparameter adjustment is crucial for the overall effectiveness of the machine learning model. In addition to that, [43] also states that increasing the accuracy of your classification model through hyperparameter adjustment will result in more accurate predictions overall. Hence, this is where the hyperparameter tuning process of the Random Forest Algorithm will take place. Here, the best fitness value for the optimal parameter results from MWOA-RF will be applied and used.

The parameter results are inserted into the random forest classifier's n_estimators and max_features where the number of trees in the model's forest is specified by the n_estimators parameter and the max_features determines the most features that the decision tree will take into account when determining the appropriate split at each node. This option affects the Random Forest's unpredictability and can aid in preventing overfitting.

While, the n_estimators have experimented using the common values from 100, 200, 300, 400 and 500. n_estimators greater than 500 resulted in diminishing returns and increased the time complexity of the model.

When it comes to random state, the researchers utilized the integer 42, however, any integer will work besides the negative integer. As per the recommendation of [44], the random state is set to 42 to achieve consistency in training and test data sets and to avoid unpredictable behavior every time the program is being run. [45] asserted that because random_state is also a hyperparameter and may be tuned to get better results, this may occasionally produce a noticeable improvement in the model's performance.

Moreover, the models are manually tuned. In certain circumstances, it is possible to maintain the random state and all other hyperparameters constant aside from the one that needs tweaking. After inserting and utilizing the hyperparameter tuning process, Training the classifier will be the next procedure. By executing the fit() function on the classifier object, the Random Forest classifier is trained using the training features (X_train) and labels (Y_train) wherein the Random Forest classifier is trained using the provided training data by executing clf.fit(X_train, y_train) where fit is used for training a machine learning model. The classifier discovers patterns and correlations in the training features and their related labels during the training phase.

3.7 Evaluation Metric

On both the training and testing datasets, predictions are made using the trained model. To determine assessment measures like accuracy, precision, and recall for both the training and testing sets, the projected labels are compared to the actual labels. The variables train_accuracy, train_precision, train_recall, test_accuracy, test_precision, and test_recall contain the evaluation metrics.

Wherein accuracy, f-1 score, precision and recall are equivalent to:

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

$$\text{Precision} = \frac{TP}{TP+FP}$$

$$\text{Recall} = \frac{TP}{TP+FN}$$

$$\text{F-1 Score} = 2 * \frac{Precision*Recall}{Precision+Recall}$$

Where:

TP = True Positives

TN = True Negatives

FP = False Positives

FN = False Negatives

Using the sklearn library, F-1 Score and K-Fold Cross Validation was utilized. The k-fold cross-validation was set to 5 iterations. It will divide the data into folds and use accuracy as the evaluation metric. Then, the cross-validation score would be displayed per fold which would tell if the data is overfitting, or underfitting, and evaluate the performance of the classifier.

When the assessment metrics have been printed, the creation of a confusion matrix will help provide a visualization of correct and incorrect classes and to assess the performance of a classification model. According to [46], high interpretability can be obtained from confusion matrices. Thus, using sci-kit-learn, which is a library of machine learning in Python that uses the confusion_matrix() function, the researchers utilized it and accepted the real labels (y_test) and predicted labels (y_test_pred) as inputs producing a confusion matrix. Using the seaborn package and matplotlib, the resulting confusion matrix is then shown as a heatmap. The counts of true positive, true negative, false positive, and false negative predictions are displayed on the heatmap.

Table 4.1 *Comparison of K-Fold Cross Validation*

Algorithm	Fold 1 Score	Fold 2 Score	Fold 3 Score	Fold 4 Score	Fold 5 Score	Average Cross-Validation Score
MWOA-RF	1.0000	1.0000	1.0000	1.0000	0.9990	0.9998
Random Forest	1.000	1.000	0.9847	1.000	0.9745	0.9864

IV. RESULTS AND DISCUSSIONS

Based on Table 4.1 presented above, MWOA-RF has the same k-fold score from the 1st fold to 4th fold with a 1.000 score and the only different score is the 5th fold with 0.9990 k-fold score.

The overall average cross-validation score for the proposed MWOA-RF is 0.9998. However, the traditional Random Forest scored 1.000 on the fold 1, 2, and 4 scores and 0.9897, and 0.9745 on folds 3 and 5 respectively. The overall average cross-validation score for the traditional RF is 0.9864.

The results suggest that MWOA-RF is slightly more stable compared to the latter though RF itself is robust. It can also be deduced that the MWOA-RF performs well across different parts of the dataset.

The fold 5 score of Random Forest suggests that there could be a potential for minimal overfitting as the fold iteration progresses.

Table 4.2 *Comparison Results of Accuracy on Proposed MWOA-RF and Traditional Random Forest*

Algorithm	Accuracy	max_features mtry	n_estimator (ntree)
MWOA-RF	97.14%	11	213
Random Forest	94.73%	7	300

As shown in Table 4.2, the proposed MWOA-RF produced an accuracy of 97.14% using the max_features of 11 and n_estimators of 213 which achieved a better result compared to the Traditional RF algorithm. On the other hand, the traditional RF produced an accuracy of 94.73% using the common default max_features and n_estimators which is lower yet still a decent accuracy in terms of classification. This suggests that the MWOA-RF with its properly tuned hyperparameters is more suitable in performing classification tasks with large datasets.

Table 4.3 *Comparison Results of F1-Score on Proposed MWOA-RF and Traditional Random Forest*

Algorithm	Precision	Recall	F1-Score
MWOA-RF	1.000	0.9580	0.9785
Random Forest	1.000	0.9226	0.9598

It can be seen in Table 4.3 that the precision scores for MWOA-RF and traditional RF were equal with the value of 1.000, while the recall scores for the proposed algorithm and traditional RF were 0.9580 and 0.9226, respectively. This implies that the proposed algorithm demonstrated a lower rate of false negatives classified and was more effective in correctly detecting positive instances. Due to that, the features chosen for the proposed MWOA-RF are preferable in demonstrating the relevance of the features or the predictors.

The F1-Score given by the MWOA-RF is 0.9785 compared to the F1-Score of the traditional RF, which is 0.9598, that signifies the proposed MWOA-RF performs better for classification and is the better model of the two since its value is closer to 1.

V. CONCLUSIONS AND RECOMMENDATIONS

5.1 Conclusions

The Random Forest Algorithm is considered prone to overfitting, improper tuning in its hyperparameters and problems for optimization, and feature selection relevance to the classifier leading it to suboptimal classification and performance. To address these existing problems on Random Forest, the researchers enhanced the following:

a) Common default parameters of the existing algorithm were not optimal when using large datasets. Hence, the researchers introduced the modified whale optimization algorithm, set the fitness function, and integrated it with the accuracy module of the Random Forest Classifier, which is capable of finding the optimal hyperparameters to properly tune the algorithm, which minimizes and solves the overfitting which shown in the k-fold cross-validation method.

b) The suboptimal performance in terms of the classification of the existing algorithm is also caused by its hyperparameters, especially when it is too high or too low. To address this, the researchers used the proposed algorithm with its modified update equation to search for the best n_estimators and max_features. It improved the classification accuracy and performance of the existing algorithm. Additionally, it solves the time complexity of trial and error when using the existing algorithm.

c) Furthermore, the existing algorithm features have the potential to not show the actual relevance of its features to the classifier. Thus, the researchers utilized the proposed algorithm to search for the best choice of features which showed optimum results through its F-1 Score, precision, and recall that suggests it can effectively classify the normal and malicious instances from the dataset binary classifier using the given set of features.

5.2 Recommendations

The researchers recommend the use of the proposed algorithm for other problems or data that need classification tasks, anomaly detection, and feature selection. In addition, the researchers recommend using the proposed algorithm on multiple large datasets to test its performance further. Also, the researchers suggest using other optimization techniques and approaches to develop the algorithm and evaluate it against those existing algorithms that are adept at handling classification tasks.

REFERENCES

[1] Farhat, R., Mourali, Y., Jemni, M., & Ezzedine, H. (2020). An overview of machine learning technologies and their use in e-learning. 2020 International Multi-Conference on: "Organization of Knowledge and Advanced Technologies" (OCTA).https://doi.org/10.1109/octa49274.2020.9151758 J.

[2] Wolff, J. (2021). How Is Technology Changing the World, and How Should the World Change Technology? https://online.ucpress.edu/gp/article/2/1/27353/118411/How-Is-Technology-Changing-the-World-and-How

[3] Zou, R., Schonlau, M. (2020) The random forest algorithm for statistical learning https://journals.sagepub.com/doi/full/10.1177/1536867X20909688

[4] Mohapatra, N., Shreya, K., Chinmay, A.(2020). Optimization of the Random Forest Algorithm. https://www.researchgate.net/publication/338561346_Optimization_of_the_Random_Forest_Algorithm

[5] He, B., Lai, S.H., Mohammed, A.S., Sabri, M.M., Ulrikh, D.V. (2022). Estimation of Blast-Induced Peak Particle Velocity through the Improved Weighted Random Forest Technique https://www.mdpi.com/2076-3417/12/10/5019

[6] Donges, N. (2021). Random Forest: A Complete Guide for Machine Learning. https://builtin.com/data-science/random-forest-algorithm

[7] Ellis, C. (2021). Random forest overfitting. Crunching the Data. https://crunchingthedata.com/random-forest-overfitting

[8] Gulati, H. (2022). Hyperparameter Tuning in Decision Trees and Random Forests. https://www.section.io/engineering-education/hyperparmeter-tuning

[9] Płoński, P. (2019). Does Random Forest overfit? https://mljar.com/blog/random-forest-overfitting

[10] Lu, Z. (2020). Enhanced Accuracy Enabled by Particle Swarm Optimization in Classification Application. 2020 International Conference on Artificial Intelligence and Computer Engineering (ICAICE). https://doi.org/10.1109/ICAICE51518.2020.00034

[11] Williams, B., Cremaschi, S. (2021). Selection of surrogate modeling techniques for surface approximation and surrogate-based optimization. https://www.sciencedirect.com/science/article/pii/S0263876221001465

[12] Meltzer, R. (2021). What Is Random Forest? https://careerfoundry.com/en/blog/data-analytics/what-is-random-forest/

[13] ProtonAutoML (2021). How to Avoid Overfitting When Using a Random Forest? https://protonautoml.medium.com/how-to-avoid-overfitting-when-using-a-random-forest-f2b900857160

[14] Verhoeven, G. (2022). Optimal performance with Random Forests: does feature selection beat tuning? https://gsverhoeven.github.io/post/random-forest-rfe_vs_tuning

[15] Williamson, B.D., Han, S., Fong, Y. (2021). Improving random forest predictions in small datasets from two-phase sampling designs. https://bmcmedinformdecismak.biomedcentral.com/articles/10.1186/s12911-021-01688-3

[16] Gupta, S., Chug, A. (2020) Software maintainability prediction using an enhanced random forest algorithm https://www.tandfonline.com/doi/abs/10.1080/09720529.2020.1728898

[17] Elavarasan, D., Vincent, P.M.D.R (2021). A reinforced random forest model for enhanced crop yield prediction by integrating agrarian parameters https://link.springer.com/article/10.1007/s12652-020-02752-y

[18] Lin, R., Pei, Z., Ye, Z., Guo, C., & Wu, B. (2019) Hydrogen fuel cell diagnostics using random forest and enhanced feature https://www.sciencedirect.com/science/article/abs/pii/S0360319919339382

[19] Koehrsen, W. (2018). Hyperparameter Tuning the Random Forest in Python. Medium https://towardsdatascience.com/hyperparameter-tuning-the-random-forest-in-python-using-scikit-learn-28d2aa77dd74

[20] Nair, G., Bai, M., Kumar, P.(2021). An efficient classification framework for breast cancer using hyperparameter tuned Random Decision Forest Classifier and Bayesian Optimization https://www.sciencedirect.com/science/article/abs/pii/S1746809421002792

[21] Chen, L. (2021). A beginner's guide to understanding and performing hyperparameter tuning for Machine Learning models https://towardsdatascience.com/the-what-why-and-how-of-hyperparameter-tuning-for-machine-learning-models-1a2634e9ca9e

[22] Zhu, Y., Xu, W., Luo, G., Wang, H., Yang, J., & Lu, W. (2020). Random Forest Enhancement using improved Artificial Fish Swarm for the medical knee contact force prediction.

[23] Esmaeili, H., Hakami, V., Bidgoli, B., Shokouhifar, M. (2022) Application-specific clustering in wireless sensor networks using combined fuzzy firefly algorithm and random forest https://www.sciencedirect.com/science/article/abs/pii/S0957417422014804

[24] Castillo, D. (2021). Machine Learning Optimisation Why is it so Important? https://www.seldon.io/machine-learning-optimisation

[25] Secherla, S. (2021). Understanding Optimization Algorithms in Machine Learning. https://towardsdatascience.com/understanding-optimization-algorithms-in-machine-learning-edfdb4df766b

[26] Sahu, P. (2022). Optimization Essentials for Machine Learning. https://www.analyticsvidhya.com/blog/2022/10/optimization-essentials-for-machine-learning

[27] Sun, W., Wang, J., Wei, X. (2018). An Improved Whale Optimization Algorithm Based on Different Searching Paths and Perceptual Disturbance https://www.mdpi.com/2073-8994/10/6/210

[28] Mohammed, H.M., Umar, S.U., Rashid, T.A. (2019). A Systematic and Meta-Analysis Survey of Whale Optimization Algorithm. https://arxiv.org/abs/1903.08763

[29] Manohar, T.G., Reddy, V.C., Reddy, P.D. (2017). Whale optimization algorithm for optimal sizing of renewable resources for loss reduction in distribution systems. https://link.springer.com/article/10.1186/s40807-017-0040-1

[30] Yan, Z., Zhang, J., Zheng, J., Tang, J. (2021). Nature-inspired approach: An enhanced whale optimization algorithm for global optimization. https://www.sciencedirect.com/science/article/abs/pii/S0378475420304638

[31] Shang, Y., Sun, G., Yuan, K., Gao, H. (2022). An Improved Whale Optimization Algorithm Based on Nonlinear Parameters and Feedback Mechanism. https://link.springer.com/article/10.1007/s44196-022-00092-7

[32] Ning, G., Cao, D. (2021). Improved Whale Optimization Algorithm for Solving Constrained Optimization Problems https://www.hindawi.com/journals/ddns/2021/8832251

[33] Guo, W., Liu, T., Dai, F., Xu, P. (2020) An Improved Whale Optimization Algorithm for Feature Selection. https://www.techscience.com/cmc/v62n1/38116

[34] Gharehchopogh, F.S., Gholizadeh, S. (2019). A comprehensive survey: Whale Optimization Algorithm and its applications. https://www.sciencedirect.com/science/article/abs/pii/S2210650218309350

[35] Rana, N., Latiff, M., Abdulhammid, S., Chiroma, H.(2020). Whale optimization algorithm: a systematic review of contemporary applications, modifications, and developments https://link.springer.com/article/10.1007/s00521-020-04849-z

[36] Mafarja, M., Mirjalili, S. (2017) Hybrid Whale Optimization Algorithm with simulated annealing

for feature selection sciencedirect.com/science/article/abs/pii/S092523121730807

[37] Nova. (2023). Improving Machine Learning Models with One-Hot Encoding. https://aitechtrend.com/improving-machine-learning-models-with-one-hot-encoding/

[38] Moustafa, N. (2017). Designing an online and reliable statistical anomaly detection framework for dealing with large high-speed network traffic. University of New South Wales, Canberra, Australia,https://www.sciencedirect.com/science/article/pii/S1877050923000807/pdf?md5=c677e4187928c832b6946f91681f6ce9&pid=1-s2.0-S1877050923000807

[39] Gimenez, L. (2020). 6 steps for data cleaning and why it matters. https://www.geotab.com/blog/data-cleaning

[40] Barkved, K. (2022). The Difference Between Training Data vs. Test Data in Machine Learning. https://www.obviously.ai/post/the-difference-between-training-data-vs-test-data-in-machine-learning

[41] Calvet et. al. (2017). Learnheuristics: hybridizing metaheuristics with machine learning for optimization with dynamic inputs. https://www.degruyter.com/document/doi/10.1515/math-2017-0029/html

[42] Vadapalli, P. (2022). Random Forest Hyperparameter Tuning: Processes Explained with Coding https://www.upgrad.com/blog/random-forest-hyperparameter-tuning

[43] Meinert, R. (2019). Optimizing Hyperparameters in Random Forest Classification https://towardsdatascience.com/optimizing-hyperparameters-in-random-forest-classification-ec7741f9d3f6

[44] Kharwal, A. (2020). Why Random_state=42 in Machine Learning? https://thecleverprogrammer.com/2020/12/17/why-random_state42-in-machine-learning

[45] Pramoditha, R. (2022) Why do we set a random state in machine learning models? https://towardsdatascience.com/why-do-we-set-a-random-state-in-machine-learning-models-bb2dc68d8431

[46] Maharaj, S. (2021) Machine Learning Model Evaluation https://www.analyticsvidhya.com/blog/2021/05/machine-learning-model-evaluation