An Improved Genetic Algorithm Based Feature Selection Technique for Intrusion Detection in Fog Computing Environment

Yusuf Muhammad Haruna
Deptment of Computer Science
Ahmadu Bello University
Zaria, Nigeria
muhdyusufharuna@gmail.com

Mohammed Abdullahi
Deptment of Computer Science
Ahmadu Bello University
Zaria, Nigeria
abdullahilwafu@abu.edu.ng

Sahabi Ali Yusuf
Deptment of Computer Science
Ahmadu Bello University
Zaria, Nigeria
sahabiali@yahoo.com

Abdulrazaq Abdulrahim
Deptment of Computer Science
Ahmadu Bello University
Zaria, Nigeria
aabdulrahim@abu.edu.ng

Usman Mohammed Joda
Department of Information Technology,
Bayero University, Kano
umjoda@gmail.com

***Abstract*— Feature Selection (FS) plays a crucial role in reducing data dimensionality, particularly in the context of Intrusion Detection (ID) with its large datasets and data imbalance challenges. Genetic Algorithm (GA) is a common choice for FS, but it suffers from slow and premature convergence to suboptimal solutions, mainly due to its limited exploitation capability, leading to diminished performance. This study introduces an improved GA-based FS technique along with a more promising machine learning algorithm designed to address data imbalance. Benchmark datasets from the Security Knowledge Discovery Dataset (NLS-KDD) are utilized to assess the proposed model, which demonstrates superior performance in terms of F1-score, precision, and recall when compared to the existing system, albeit with a slight advantage for the existing system in performance accuracy and execution time.**

Keywords—Fog computing, Intrusion detection system, genetic algorithm, AdaBoost.

I. INTRODUCTION

The rise in network attacks, driven by increased internet usage, is a major concern in cybersecurity [1]. Intrusion Detection Systems (IDS) play a crucial role in network security by identifying and preventing intrusions to safeguard network availability, integrity, and confidentiality [2]. Fog computing supplements cloud computing to offer low-latency services and local processing, making it vital to ensure network communication legitimacy on fog devices [3][4].

Research on Intrusion Detection (ID) in fog computing often employs machine learning, deep learning, and ensemble-based detection techniques [5]. Metaheuristic algorithms, such as Genetic Algorithm (GA), are used to address real-world issues in various fields [6]. These algorithms balance exploitation and exploration and can be single-solution or population-based [7]. GA, based on genetic and evolutionary concepts, uses chromosome representation and biologically inspired operators [8][9].

However, GA faces issues like slow convergence to suboptimal solutions due to its weak exploitation. Researchers have proposed various mutation operators, including scramble and bit flipping mutations, to strengthen GA's exploitation ability. Scramble mutation shuffles a subset of genes in the solution vector but can disrupt population quality. Bit flipping mutation randomly flips a gene but may not guarantee significant improvements, especially for large-scale problems. To enhance GA's exploitation ability in feature selection, a hybrid of scramble and bit flipping mutations is proposed.

II. BACKGROUND

This section gives a summary of the study's major concepts and discusses recent literal works that use those terms, such as imbalance data, genetic algorithm, intrusion detection system and adaboost.

A. Imbalance data

Data imbalance occurs when certain class variables have significantly fewer instances compared to others, leading to inaccurate model representations in classification [10][11]. This issue is common in various fields, such as environmental data, fraud detection, network intrusion detection, and medical diagnostics [12]. To address data imbalance, two main approaches are used: algorithm-driven and data-driven methods [12].

1) Data-Driven Approach

Data-driven approaches, such as resampling techniques, aim to balance class distribution in imbalanced datasets [10]. Under-sampling randomly removes majority class examples, potentially losing crucial information, while random oversampling replicates minority class instances, risking classifier overfitting and increased computational costs [10] [13]

2) Algorithm Driven Approach

This method, described by [10], adjusts classification algorithms to better handle imbalanced data for minority classes. It employs cost-sensitive learning, where higher costs are imposed on classifiers for misclassifications, particularly crucial in sensitive areas like medical diagnostics. Thresholding techniques address class misclassification issues algorithmically. Additionally, hybrid methods, such as ensemble learning, specifically AdaBoost, combine classifiers to reduce variance and enhance prediction accuracy.

AdaBoost, as outlined by [13], involves up and down sampling combinations with dynamically adjusted weights for misclassified instances to improve overall classifier performance.

B. Intrusion Detection System(IDS)

An Intrusion Detection System (IDS) is a software tool that monitors a network for malicious activities and policy violations. It automatically detects and classifies intrusions, attacks, or security policy breaches at both the network and host levels [14]. Intrusion detection is categorized into host-based (HIDS) and network-based (NIDS) systems, with NIDS focusing on network behavior analysis using network equipment like switches and routers, while HIDS evaluates information on single or multiple host systems [14] [15]. The four basic types of attacks are, denial of service, probe, user to remote, and user to root attacks.

C. Dimentionality Reduction

Dimensionality reduction (DR) is a preprocessing step focused on reducing training time and enhancing learning accuracy by eliminating noise [16]. DR includes two main types: feature extraction and feature selection [16] [17].

D. Feature Selection

Feature selection (FS) involves choosing the most crucial features from available data for tasks like classification, regression, or clustering [18]. FS offers several benefits, including reduced data size, enhanced classification accuracy, minimized storage space, and decreased computational/training time by simplifying variable understanding [19]. Overall, FS has the potential to improve performance, readability, and interpretability of models, particularly when dealing with large feature sets or aiming for a generalized model [20].

E. Wrapper method

The wrapper method selects feature subsets using a predefined learning algorithm, evaluating the model's error rate on a test set. The chosen subset, with the highest score, is selected, but this method is computationally intensive. The sequential approach, commonly used in wrapper techniques, involves starting with a blank or full set and incrementally adding or removing features until termination requirements are met. Sequential Backward Selection (SBS) removes irrelevant features from a full set, while Sequential Forward Selection (SFS) adds features sequentially based on relevance [21].

F. Metaheuristic Algorithms

[22] formally defined a metaheuristic as an iterative process guiding a subordinate heuristic to explore (Diversification) and exploit (Intensification) a search space by merging various ideas. Metaheuristics, broad algorithmic frameworks, employ learning processes to quickly identify near-optimal solutions for various optimization problems with minimal modifications [23]. Balancing intensification and diversification, metaheuristic algorithms provide fast and effective solutions for NP-hard issues with numerous variables and non-linear objectives [24].

G. Genetic Algorithm

Genetic Algorithm (GA), inspired by natural selection, is a population-based optimization method employing survival-of-the-fittest principles [25]. GA involves key elements like chromosome representation, selection, crossover, mutation, and fitness function calculation. Selection, a crucial aspect, determines which strings participate in reproduction, influencing the convergence rate; common methods include roulette wheel, rank, tournament, boltzmann, and stochastic universal sampling. Crossover combines genetic material from parents to generate offspring, employing operators like single-point, two-point, k-point, uniform, and others. Mutation introduces randomness by altering gene values, preventing identical solutions and avoiding local optima. Popular mutation operators include displacement, simple inversion, and scrambling. Termination strategies, crucial for ending the search, include a predetermined number of generations, fulfillment of minimum criteria, and plateau recognition [26] [27] [28].

H. Ensemble

Ensemble learning, a crucial branch of machine learning [29], addresses classification problems by combining multiple weak classifiers into a robust classifier, enhancing generalizability and robustness compared to using a single estimator [30]. The key concept is leveraging the limitations of individual weak classifiers by merging them to create a powerful and stable classifier. In ensemble learning algorithms like Boosting, weak classifiers in the first type, such as Boosting, have mutual dependencies, while in the second type, like Bagging and Random Forest, weak classifiers are independent [31]. Boosting involves training weak classifiers sequentially, adjusting sample distribution based on previous results, and combining the weights of these classifiers to generate a strong classifier after a specified number of iterations [31].

I. AdaBoost

Introduced by [32], the Adaboost algorithm is a powerful ensemble learning method that enhances diagnostic accuracy with unbalanced datasets. It sequentially trains weak classifiers, adjusting data distribution by assigning equal weights initially and updating weights based on classification accuracy in each iteration. The final model is a linear combination of base learners weighted by their individual performance [33] [34] [35] [36].

J. Related Works

Fog computing as a fast-developing research field has so far increased the devotion of some researchers considering its low cost, performance, and accuracy in processing data in low latency. So far, some scholarly work has been documented in this work. [37] a detailed investigation was done into a number of machines learning techniques, including Decision Tree, KNN, Random Forest, and others, as well as how machine learning-based intrusion detection tactics at the fog layer might detect anomalies or attacks. [38] proposed an Intrusion Detection System based on a decision tree, the technique not only recognizes four types of attack but also makes twenty-two

types of attacks detectable, the IDS system primarily consists of three parts: data preprocessing, data normalization, and decision tree detection. However, the problem of imbalanced data has not been addressed in any of these steps. [39] also proposed a set of anomaly detection algorithms for fog environments that categorize IoT traffic as either normal or malicious, and then send that traffic to the cloud for malicious attack mitigation, in comparison to an existing centralized deep learning system, this method can handle scaled data by resolving the class imbalance issue, the detection accuracy of each class for multiclass classification may be increased. Other works carried on in the same vein include that of [40] that proposed a method (Auto-If) for ID using autoencoder (AE) and isolation forest (IF) for the fog environment, this method only focuses on binary classification and treats each dataset characteristic as equally significant. [1] designed a genetic algorithm-based feature selection and Naïve Bayes for anomaly detection in a fog computing environment which much emphasis wasn't given to address the issue of imbalanced data, as a result, the model can be biased to the classification of the attack, though their results recorded 99.73% accuracy, with a false positive as low as 0.6%.

Ensemble methods are a subset of machine learning algorithms that have piqued the scientific community's curiosity. This curiosity is well-deserved, as ensemble learning techniques have shown to be quite powerful and quite useful in a range of applications. As a result, various intrusion detection algorithms based on ensemble learning have been adopted. [42] employed a Bayesian network and random tree as two basic classification approaches to tackle the difficulty of an unbalanced data sample. The authors provide a new intrusion dataset, the KDDcup99 dataset, which is used to evaluate the proposed ensemble model. Artificially enlarging the dataset through label-preserving modifications is the simplest and most frequent strategy for overcoming restricted datasets and reducing over-fitting of deep learning models for picture classification. A semi-supervised version of Adaboost, one of the most useful ensemble-based algorithms, was used to detect network anomalies [42]. [35] propose PwAdaBoost, a possible world-based AdaBoost method. According to the authors, the provided method is can manage various kinds of uncertain data and, as a result, it may be utilized in order to handle ambiguity in intrusion analysis. In [43] introduced an adaptive ensemble machine learning method that has been used to identify network assaults including DoS, Probe, U2R, and R2L. To increase classification accuracy, several classifiers, including decision trees, random forests, KNNs, and deep neural networks (DNNs), have been implemented. The results of the simulations reveal that the suggested technique outperforms other learning algorithms. To obtain high classification accuracy. [44] apply the AdaBoost algorithm for fault diagnosis. The suggested AdaBoost-based classification approach has good fault diagnostic accuracy, according to simulation findings [45] suggest using the AdaBoost algorithm for detecting low-rate denial of service (LDoS) attacks in communication networks, an enhanced version of the AdaBoost method called MF-Adaboost is developed and implemented by the authors for that reason, as well as define a set of network traffic aspects that are the most useful classification features. The simulation results suggest that the proposed strategy is effective in detecting LDoS assaults.

III. METHODOLOGY

In this section, we describe the research method utilized in the study. It provides information on the parameters and settings used in the proposed technique, the benchmark datasets, the system specification, the hybrid method and the experimental designs.

A. Dataset Description

In this analysis, the NSL-KDD benchmark dataset is used by the proposed model as evaluation data. This dataset has been utilized as a standard dataset by numerous researchers in the field of IDS. It uses three different sets of data: the entire dataset, the 20% of entire dataset for training, and the full KDD testing dataset. A set of 41 attributes and a label classifying each record-specific attack or normal are used to define each instance. Tables I and II show, respectively, the total number of normal and attack cases in each fold of testing and training data.

TABLE I. NUMBER OF ATTACK CASES IN THE TRAINING DATASET

Types of attack	Number of records
Normal	67343
DoS	45927
Probe	11656
R2L	995
U2R	52
Total	**125973**

TABLE II. NUMBER OF ATTACK CASES IN THE TRAINING DATASET

Types of attack	Number of records
Normal	9711
DoS	7456
Probe	2421
R2L	2756
U2R	200
Total	**22544**

B. Proposed Techique

In scramble mutation, a subset of genes is selected from the solution vector (must not be contiguous) and their corresponding values are shuffled or scrambled randomly. Although the scramble mutation affects large number of genes which makes it most suitable for large scale problems introducing exploration, it has the tendency of causing disturbance in the population leading to deterioration of solution quality in some problems.

In the bit flipping mutation, a random gene is selected from the solution vector base on some probability and is flipped i.e., from 0 to 1 and vice versa, with the aim of creating different solutions with new genes differing from those in the parents. Although the bit flipping mutation has proven to be effective offering exploitation, there is no guarantee of significant improvement and is not practical or sufficient when applied to large scale problem.

Algorithm 1: Improve Genetic Algorithm

Input: Dataset; Population size: N; Maximum number of iteration: M_{max}
Output: Class labelled test instance

Randomly generate initial solution X(0)
For $K = 1$ to N do
  Evaluate fitness of individual of X(0);
End for
$m = 0$
While $X(m \leq M_{max})$ do
  Select two individuals in X(0) based on their fitness
  Apply crossover operator to selected individual from step 3 resulting to new individuals
  Apply scramble mutation operator to new individuals from step 5
  Evaluate the fitness of individual of X(t)
  If fitness deteriorates then
    Select random gene from the initial subset of gene
    Flip the selected gene based on some probability
    Re-evaluate the fitness of individual of X(t) after bit flipping
    If fitness improves then
      Replace original individual X(t) with the mutated one
    End if
  End if
  $t = t + 1$;
End While
Return individual with highest fitness

Therefore, we propose to create a hybrid of the scramble and bit flipping mutations to improve on the exploitation ability of GA in avoiding slow and premature convergence to suboptimal solutions in feature selection problem. The working of the proposed hybrid method is as follows:

1: For mutation, apply scramble mutation operator. After applying scramble mutation operator, two new solution vectors are generated say $C1$ and $C2$.

2: Fitness values of the generated solution vectors $C1$ and $C2$ are calculated

3: If fitness of either solution vector deteriorates or fails to improve, select a random bit (gene) based on some probability from the initial subset of genes selected in (1) and flip.

4: Re-calculate the fitness value of the newly created solutions say $C1$bf and $C2$bf after bit flipping.

5: If fitness of either $C1$bf and $C2$bf is better than $C1$ and $C2$, replace solution vectors with new solution,

6: Else, return to initial population.

Algorithm 1, and Figure 1 depicts the proposed hybridization

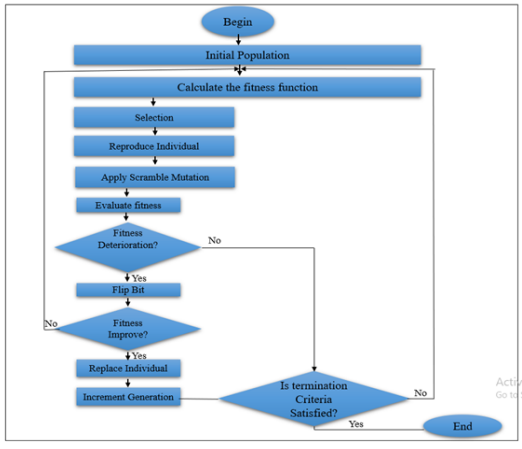

Fig. 1. Flow chart of the proposed Hybridization

C. *Classifier (AdaBoost)*

The AdaBoost algorithm selects m groups of training data at random from the sample space, initializes the test data distribution weight $D_1 = 1/m$, and obtains the first base classifier h1 using the initial data distribution; the tth base classifier is constructed iteratively, utilize the classifier's overall prediction error ε_t, and discover a zero solution for the exponential loss function's $\ell_{\exp}(\beta_t|D_t)$ partial derivative (2) to determine the classifier's weight β_t.

$$e_t = \sum_i D_i(i) \ (if \ h_t(x_i) \neq y_i) \quad (1)$$

$$\frac{\delta l(\beta_t|D_t)}{\delta \beta_t} = -e^{-\beta_t}(1-\varepsilon_t) + e^{\beta_t}\varepsilon_t \quad (2)$$

$$\beta_t = \frac{1}{2}\ln\left(\frac{1-e_t}{e_t}\right) \quad (3)$$

Using the pertinent parameters learned during the t^{th} base classifier training, adjust the weights of the test data D_{t+1} for the $t+1^{th}$ iteration, The normalizing factor is β_t, y_i represents the actual classification outcome, y_i^t represents the tth base classifier's predicted classification results.

Algorithm 2: AdaBoost Algorithm (Long et al., 2021)

Given $(x_1, y_1), \ldots, (x_m, y_m)$ *Where* $x_i \in X, y_i \in \{-1, +1\}$.
Initialize: $D_1 = 1/m$ *for* $i = 1, \ldots, m$.
For $t = 1, \ldots, T$:
1 *Train weak learner using distribution* D_t.
2 *Get weak hypothesis* $x \rightarrow \{-1, +1\}$.
3 *Aim: select* h_t *with low weighted error:*

$$e_t = \sum_i D_t(i) \ (if \ h_t(x_i) \neq y_i)$$

4 *Choose* $\beta_t = \frac{1}{2}\ln\left(\frac{1-e_t}{e_t}\right)$
5 *Update, for* $i = 1, \ldots m$:

$$D_{t+1}(i) = \frac{D_t(i)}{\beta_t} * \exp(-\beta_t y_i y_i^t)$$

Where β_t *is a normalization factor (chosen so that* D_{t+1} *will be a distribution).*
Output the final hypothesis:

$$H(x) = sign\left(\sum_{t=1}^{T} \beta_t h_t(x)\right)$$

$$D_{t+1}(i) = \frac{D_t(i)}{\beta_t} * \exp(-\beta_t y_i y_i^t) \quad (4)$$

Greater weight is given to the fundamental classifier with a low classification error rate, and a lower weight value is assigned to the basic classifier with a high classification error rate. Following that, the linear combination based on T base classifiers is obtained.

$$f(x) = \sum_{t=1}^{T} \beta_t h_t(x) \quad (5)$$

On the basis of linear combination, the sign function is transformed to get the result of strong classifier.

$$H(x) = sign(f(x)) = sign\left(\sum_{t=1}^{T} \beta_t h_t(x)\right) \quad (6)$$

D. Evaluation Metrics

On the basis of what is known as the confusion matrix, some metrics are used to assess the performance of classification models. The matrix is N x N, where N is the total number of target values (classes), and it displays the proportion of accurate and inaccurate predictions made by the predictive model in relation to the data's actual results. Among the entries in the confusion matrix are True positive (TP) the number of instances that were both truly positive and predicted by the classifier, True negative (TN) the number of instances that were expected to be negative and actually turned out to be negative, False positive (FP) the number of occurrences where a result was projected to be negative but turned out to be positive, False negative (FN) the number of occurrences where a result was projected to be positive but turned out to be negative.

Accuracy, precision, recall, F1-score, and execution time are used as performance metrics in this study. These metrics are defined as follows:

1) *Accuracy:* is the percentage of total predictions that were accurate is known as accuracy. The following formula can be used to obtain accuracy:

$$Accuracy = \frac{TP*TN}{TP*TN*FP*FN} \quad (7)$$

2) *Precision (P):* Is the number of pertinent instances among the discovered instances. The following formula can be used to calculate R:

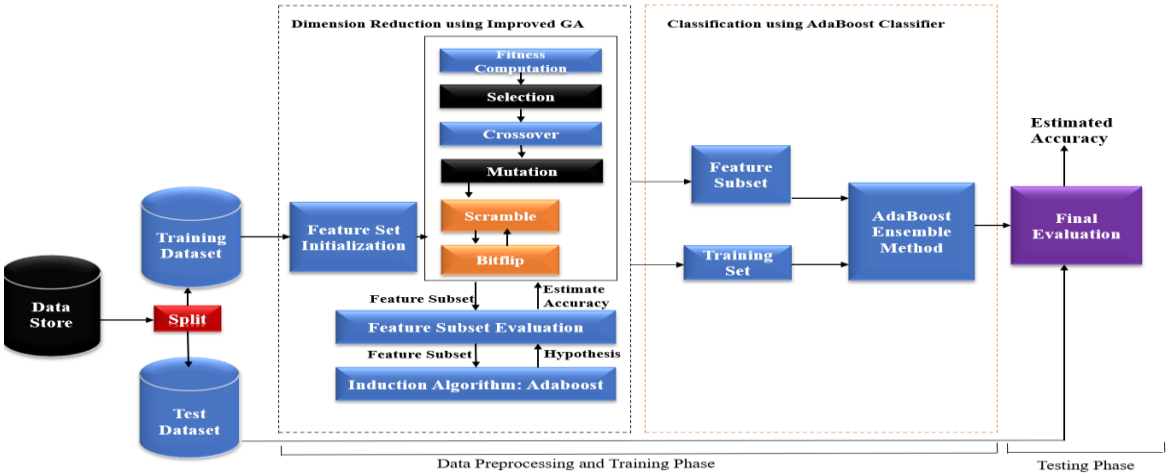

Fig. 2. Model Framework

$$Precision = \frac{TP}{TP+FP} \quad (8)$$

3) Recall (R): Is the percentage of relevant occurrences that have been found over all relevant instances. The following formula can be used to calculate R:

$$Recall = \frac{TP}{TP+FN} \quad (9)$$

4) *F1 score* is a harmonic mean of precision and recall, which represents the class's weighted average in terms of accuracy and can be obtained using the following formula:

$$F1\ score = \frac{2PR}{P+R} \quad (10)$$

5) The excution time of the suggested method is contain ed in the computation time t of the IDS detection algo rithm.

E. Experimental Setup

All the experiments were carried out on a PC with an Intel(R) Core TM i5-2410M CPU running at 2.4 GHz with 4.00 GB of RAM, and 64-bit Windows 10. The experiment was conducted using the Python programming language in a Jupyter notebook.

F. Model Building

As part of the first step of the wrapper feature selection strategy, the Genetic Algorithm (GA), a random selection method as indicated in Algorithm 3.3, has been used as a feature search algorithm during the data pre-processing and training phases of the proposed model. GA is a type of random search that is effective in finding huge search areas, which is often essential in feature selection.

Additionally, GAs does a global search as opposed to a local or greedy search, unlike many other search algorithms. The core idea is to evolve a population of individuals, each of whom could be a potential answer to a particular problem. Reproduction, crossover, and mutation are the three main operators in a genetic algorithm [1].

An initial population of potential solutions is seeded randomly by the genetic algorithm. The chromosomal genes are represented by bits, characters, or numbers depending on the nature and characteristics of the NSL-KKD dataset. On the basis of their fitness function, individuals are then assessed. Following that, the entire population is subjected to the selection and recombination operators, each with a probability

of 0.6, in order to discover fresh approaches in the search space. The solutions are then improved by random modification using the scramble and bit flipping mutation operators, which has a probability of 0.033. As demonstrated in Algorithm 1, at the conclusion of the dimension reduction, the NSL-features dataset was reduced to 17. The wrapper technique needs a classification algorithm, to assess how well the feature subset performs. The AdaptiveBoost classifier is used in this study to accomplish both of these goals, as well as to categorize the attacks using the chosen features. Figure 2 depicts the operation of the proposed model.

IV. Results and Discussion

The experiment was conducted on a subset and entire of the world-renowned reference point dataset NSL-KDD, which housed 25,192 cases of 41 features, and in addition to the basic type of class label, four additional attack types known as Probing, DoS, R2L, and U2R. Because k-fold cross-validation is a well-known method for carrying out any classification system because it reduces the possibility of developing an over-fitted classification model, all of the experiments were carried out using k-fold cross-validation.

A. The model performance

In this experiment, the performance of the model was investigated using some selected hyperparameters.

Table III. Selected hyperparameters for the experiment

Learning _rate	0.6
n_ estimator	1000
Random _state	42
Base _estimator	Decision Tree

The model recorded the best accuracy with a score of accuracy, precision, recall, F1-score, and time taken with scores of 99.72%, 99.70%, 99.74%, 99.69%, and 28sec respectively which is almost the same as the result obtained from our base work in terms of accuracy. However, recorded the best F1-score of 99.69% which is higher than that of our base work as can be seen in Table IV below.

Table IV. Performance of the proposed system subset dataset

Measure	20% of NLS-KDD
Accuracy	99.72
Precision	99.70
Recall	99.74
F1-Score	99.69
Time Taken (sec)	28

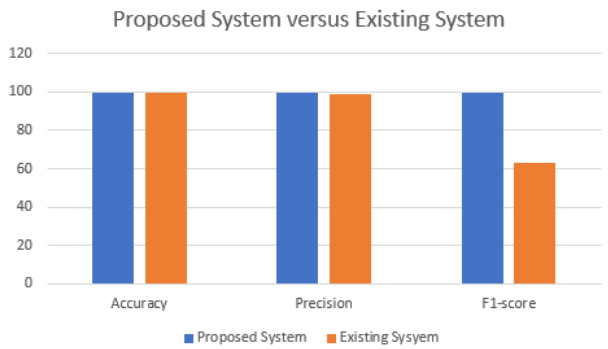

Fig. 3. Comparison of the performance metric of the proposed versus existing system

B. Discussion

The results obtained from the experiment above, the proposed model has beaten the existing model in terms of f1-score and precision, according to the findings shown in Table IV. The capacity of the model to use the improved GA to escape the local optima is what makes it perform better than the existing model. In comparison to the existing model, the proposed model chooses fewer features while maintaining a higher classification accuracy. As depicted in Figure 3.

It is worth pointing out that even though the proposed model failed to achieve the best execution time, it was able to obtain a comparative result in comparison to three of the evaluation metrics which include F1-score, accuracy, and precision. figure 4 depicts the execution time.

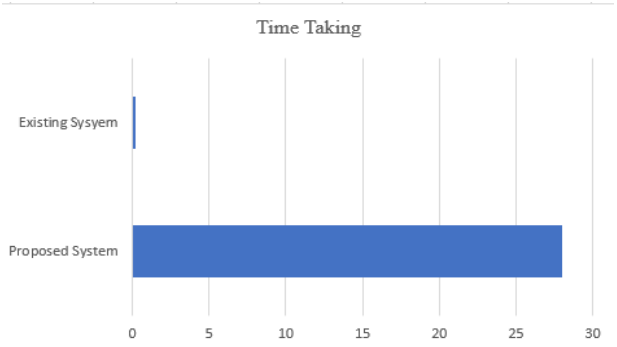

Fig. 4. Comparison of execution time of the proposed

V. Conclusion and Future Research Directions

In this research work, our focus was on Genetic Algorithms (GA) for feature selection problems and addressing data imbalance. This involved developing an updated GA that reduces the number of features while maximizing classification accuracy, taking into account the features that significantly

contribute to better classification. Subsequently, a series of experiments were conducted to determine the optimal parameters values. The hyperparameters were adjusted manually, considering experimentation results and domain knowledge. The proposed model achieved an impressive F1-score of 99.69%, outperforming existing models. The selected hyperparameters included a learning rate of 0.6 and n_estimators set to 1000. It can be concluded that the proposed solutions prove to be strong contenders for tackling imbalanced and feature selection problems.

For future work and direction, the proposed model can be further investigated using various types of datasets. Furthermore, different classifiers, in addition to AdaBoost, can be experimented with to assess their performance. Additionally, exploring other techniques for addressing imbalanced problems may lead to a reduction in the execution time of our model.

REFERENCES

[1] Oche, J., Abdulhamid, M., Abdullahi, M., Hayatu, I., & Al-ghusham, A. (2021). Machine Learning with Applications Genetic Algorithm based feature selection and Naïve Bayes for anomaly detection in fog computing environment. *Machine Learning with Applications*, *6*(September), 100156. https://doi.org/10.1016/j.mlwa.2021.100156

[2] Kumar, P., Gupta, G. P., & Tripathi, R. (2020). A distributed ensemble design based intrusion detection system using fog computing to protect the internet of things networks. *Journal of Ambient Intelligence and Humanized Computing*, *0123456789*. https://doi.org/10.1007/s12652-020-02696-3

[3] Atlam, H. F., Walters, R. J., & Wills, G. B. (2018). Fog computing and the internet of things: A review. *Big Data and Cognitive Computing*, *2*(2), 1–18. https://doi.org/10.3390/bdcc2020010

[4] Chen, C., Tsai, Y., Chang, F., & Lin, W. (2020). Ensemble feature selection in medical datasets : Combining filter , wrapper , and embedded feature selection results. October 2019, 1–10. https://doi.org/10.1111/exsy.12553

[5] Sadaf, K., & Sultana, J. (2020b). Intrusion detection based on autoencoder and isolation forest in fog computing. *IEEE Access*, *8*, 167059–167068. https://doi.org/10.1109/ACCESS.2020.3022855

[6] Kumar, V., Chhabra, J. K., & Kumar, D. (2014). Parameter adaptive harmony search algorithm for unimodal and multimodal optimization problems. *Journal of Computational Science*, *5*(2), 144–155. https://doi.org/10.1016/j.jocs.2013.12.001

[7] Katoch, S., Chauhan, S. S., & Kumar, V. (2021). A review on genetic algorithm: past, present, and future. In *Multimedia Tools and Applications* (Vol. 80, Issue 5). Multimedia Tools and Applications. https://doi.org/10.1007/s11042-020-10139-6

[8] Audet, C., & Hare, W. (2017). Genetic Algorithms. *Springer Series in Operations Research and Financial Engineering*, 57–73. https://doi.org/10.1007/978-3-319-68913-5_4

[9] Hasanova, N. (2020). A Comparative Study of Particle Swarm Optimization and Genetic Algorithm. *Qubahan Academic Journal*, *1*(1), 33–45. https://doi.org/10.48161/qaj.v1n1a7

[10] Thabtah, F., Hammoud, S., Kamalov, F., & Gonsalves, A. (2020). Data imbalance in classification: Experimental evaluation. *Information Sciences*, *513*, 429–441. https://doi.org/10.1016/j.ins.2019.11.004

[11] Gao, X., Shan, C., Hu, C., Niu, Z., & Liu, Z. (2019). An Adaptive Ensemble Machine Learning Model for Intrusion Detection. *IEEE Access*, *7*, 82512–82521. https://doi.org/10.1109/ACCESS.2019.2923640

[12] Buda, M. (2017). A systematic study of the class imbalance problem in convolutional neural networks SCHOOL OF COMPUTER SCIENCE AND COMMUNICATION A systematic study of the class imbalance problem in convolutional neural networks. 49.

[13] Ali, H., Salleh, M. N. M., Saedudin, R., Hussain, K., & Mushtaq, M. F. (2019). Imbalance class problems in data mining: A review. *Indonesian Journal of Electrical Engineering and Computer Science*, *14*(3), 1552–1563. https://doi.org/10.11591/ijeecs.v14.i3.pp1552-1563

[14] Vinayakumar, R., Alazab, M., Soman, K. P., Poornachandran, P., Al-Nemrat, A., & Venkatraman, S. (2019). Deep Learning Approach for Intelligent Intrusion Detection System. *IEEE Access*, *7*, 41525–41550. https://doi.org/10.1109/ACCESS.2019.2895334

[15] Mishra, P., Varadharajan, V., Tupakula, U., & Pilli, E. S. (2019). A detailed investigation and analysis of using machine learning techniques for intrusion detection. *IEEE Communications Surveys and Tutorials*, *21*(1), 686–728. https://doi.org/10.1109/COMST.2018.2847722

[16] Velliangiri, S., Alagumuthukrishnan, S., & Thankumar Joseph, S. I. (2019). A Review of Dimensionality Reduction Techniques for Efficient Computation. *Procedia Computer Science*, *165*, 104–111. https://doi.org/10.1016/j.procs.2020.01.079

[17] Effrosynidis, D., & Arampatzis, A. (2021). An evaluation of feature selection methods for environmental data. *Ecological Informatics*, *61*(December 2020), 101224. https://doi.org/10.1016/j.ecoinf.2021.101224

[18] Solorio-Fernández, S., Carrasco-Ochoa, J. A., & Martínez-Trinidad, J. F. (2020). A review of unsupervised feature selection methods. *Artificial Intelligence Review*, *53*(2), 907–948. https://doi.org/10.1007/s10462-019-09682-y

[19] Zebari, R., Abdulazeez, A., Zeebaree, D., Zebari, D., & Saeed, J. (2020). A Comprehensive Review of Dimensionality Reduction Techniques for Feature Selection and Feature Extraction. *Journal of Applied Science and Technology Trends*, *1*(2), 56–70. https://doi.org/10.38094/jastt1224

[20] Hancer, E., Xue, B., & Zhang, M. (2020). A survey on feature selection approaches for clustering. *Artificial Intelligence Review*, *53*(6), 4519–4545. https://doi.org/10.1007/s10462-019-09800-w

[21] Moran, M., & Gordon, G. (2019). Curious Feature Selection. *Information Sciences*, *485*, 42–54. https://doi.org/10.1016/j.ins.2019.02.009

[22] Laporte, G., & Osman, I. H. (1995). Routing problems: A bibliography. *Annals of Operations Research*, *61*(1), 227–262. https://doi.org/10.1007/BF02098290

[23] Blum, C., & Roli, A. (2003). Metaheuristics in Combinatorial Optimization: Overview and Conceptual Comparison. *ACM Computing Surveys*, *35*(3), 268–308. https://doi.org/10.1145/937503.937505

[24] Gogna, A., & Tayal, A. (2013). Metaheuristics: Review and application. In *Journal of Experimental and Theoretical Artificial Intelligence* (Vol. 25, Issue 4, pp. 503–526). Taylor & Francis. https://doi.org/10.1080/0952813X.2013.782347

[25] Michalewicz, Z., & Schoenauer, M. (1996). *Evolutionary Algorithms for Constrained Parameter Optimization Problems 1 Introduction*. *4*(March), 1–32. https://doi.org/10.1162/evco.1996.4.1.1

[26] Li, T., Shao, G., Zuo, W., & Huang, S. (2017). Genetic algorithm for building optimization - State-of-the-art survey. *ACM International Conference Proceeding Series*, *Part F1283*, 205–210. https://doi.org/10.1145/3055635.3056591

[27] Audet, C., & Hare, W. (2017). Genetic Algorithms. *Springer Series in Operations Research and Financial Engineering*, 57–73. https://doi.org/10.1007/978-3-319-68913-5_4

[28] Katoch, S., Chauhan, S. S., & Kumar, V. (2021). A review on genetic algorithm: past, present, and future. In *Multimedia Tools and Applications* (Vol. 80, Issue 5). Multimedia Tools and Applications. https://doi.org/10.1007/s11042-020-10139-6

[29] Dietterich, T. G. (2000). Ensemble methods in machine learning. Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 1857 LNCS, 1–15. https://doi.org/10.1007/3-540-45014-9_1

[30] Gao, X., Shan, C., Hu, C., Niu, Z., & Liu, Z. (2019). An Adaptive Ensemble Machine Learning Model for Intrusion Detection. *IEEE Access*, *7*, 82512–82521. https://doi.org/10.1109/ACCESS.2019.2923640

[31] Tang, D., Tang, L., Dai, R., Chen, J., Li, X., & Rodrigues, J. J. P. C. (2020). MF-Adaboost: LDoS attack detection based on multi-features

and improved Adaboost. *Future Generation Computer Systems*, *106*, 347–359. https://doi.org/10.1016/j.future.2019.12.034

[32] Freund, Y. (1995). Boosting a weak learning algorithm by majority. *Information and Computation*, *121*(2), 256–285. https://doi.org/10.1006/INCO.1995.1136

[33] Liu, Han, Zhang, X., & Zhang, X. (2019). PwAdaBoost: Possible world based AdaBoost algorithm for classifying uncertain data. *Knowledge-Based Systems*, *186*(xxxx), 104930. https://doi.org/10.1016/j.knosys.2019.104930

[34] Long, Z., Zhang, X., Zhang, L., Qin, G., Huang, S., Song, D., Shao, H., & Wu, G. (2021). Motor fault diagnosis using attention mechanism and improved adaboost driven by multi-sensor information. *Measurement: Journal of the International Measurement Confederation*, *170*, 108718. https://doi.org/10.1016/j.measurement.2020.108718

[35] Pham, B. T., Nguyen, M. D., Nguyen-Thoi, T., Ho, L. S., Koopialipoor, M., Kim Quoc, N., Armaghani, D. J., & Le, H. Van. (2021). A novel approach for classification of soils based on laboratory tests using Adaboost, Tree and ANN modeling. *Transportation Geotechnics*, *27*(July 2020), 100508. https://doi.org/10.1016/j.trgeo.2020.100508

[36] Zharmagambetov, A., Gabidolla, M., & Carreira-Perpinan, M. A. (2021). *Improved Multiclass Adaboost For Image Classification: The Role Of Tree Optimization*. *1*, 424–428. https://doi.org/10.1109/icip42928.2021.9506569

[37] Moh, M., & Raju, R. (2018). Machine learning techniques for security of internet of things (IoT) and fog computing systems. *Proceedings - 2018 International Conference on High Performance Computing and Simulation, HPCS 2018*, 709–715. https://doi.org/10.1109/HPCS.2018.00116

[38] Peng, K., Leung, V. C. M., Zheng, L., Wang, S., Huang, C., & Lin, T. (2018). Intrusion detection system based on decision tree over big data in fog environment. *Wireless Communications and Mobile Computing*, *2018*. https://doi.org/10.1155/2018/4680867

[39] Bhuvaneswari, B. A., & S., S. (2020). Anomaly detection framework for Internet of things traffic using vector convolutional deep learning approach in fog environment. *Future Generation Computer Systems*, *113*, 255–265. https://doi.org/10.1016/j.future.2020.07.020

[40] Sadaf, K., & Sultana, J. (2020b). Intrusion detection based on autoencoder and isolation forest in fog computing. *IEEE Access*, *8*, 167059–167068. https://doi.org/10.1109/ACCESS.2020.3022855

[41] Wang, Y., Shen, Y., & Zhang, G. (2016). Research on Intrusion Detection Model using ensemble learning methods. *Proceedings of the IEEE International Conference on Software Engineering and Service Sciences, ICSESS*, *0*(Cml), 422–425. https://doi.org/10.1109/ICSESS.2016.7883100

[42] Yuan, Y., Kaklamanos, G., & Hogrefe, D. (2016). A novel semi-supervised adaboost technique for network anomaly detection. MSWiM 2016 - Proceedings of the 19th ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems, 111–114. https://doi.org/10.1145/2988287.2989177