Arabian Journal for Science and Engineering
https://doi.org/10.1007/s13369-024-08949-z

RESEARCH ARTICLE-COMPUTER ENGINEERING AND COMPUTER SCIENCE

Empirical Enhancement of Intrusion Detection Systems: A Comprehensive Approach with Genetic Algorithm-based Hyperparameter Tuning and Hybrid Feature Selection

Halit Bakır[1] · **Özlem Ceviz**[2]

Received: 27 August 2023 / Accepted: 10 March 2024

Abstract
Machine learning-based IDSs have demonstrated promising outcomes in identifying and mitigating security threats within IoT networks. However, the efficacy of such systems is contingent on various hyperparameters, necessitating optimization to elevate their performance. This paper introduces a comprehensive empirical and quantitative exploration aimed at enhancing intrusion detection systems (IDSs). The study capitalizes on a genetic algorithm-based hyperparameter tuning mechanism and a pioneering hybrid feature selection approach to systematically investigate incremental performance improvements in IDS. Specifically, our work proposes a machine learning-based IDS approach tailored for detecting attacks in IoT environments. To achieve this, we introduce a hybrid feature selection method designed to identify the most salient features for the task. Additionally, we employed the genetic algorithm (GA) to fine-tune hyperparameters of multiple machine learning models, ensuring their accuracy in detecting attacks. We commence by evaluating the default hyperparameters of these models on the CICIDS2017 dataset, followed by rigorous testing of the same algorithms post-optimization through GA. Through a series of experiments, we scrutinize the impact of combining feature selection methods with hyperparameter tuning approaches. The outcomes unequivocally demonstrate the potential of hyperparameter optimization in enhancing the accuracy and efficiency of machine learning-based IDS systems for IoT networks. The empirical nature of our research method provides a meticulous analysis of the efficacy of the proposed techniques through systematic experimentation and quantitative evaluation. Consolidated in a unified manner, the results underscore the step-by-step enhancement of IDS performance, especially in terms of detection time, substantiating the efficacy of our approach in real-world scenarios.

Keywords Intrusion detection systems (IDSs) · Machine learning (ML) · Hyperparameters tuning · Genetic algorithm

1 Introduction

Due to advances in network technology, the Internet is witnessing a significant surge in connected devices and applications. The number of Internet of Things (IoT) devices skyrocketed by 18%, hitting 14.4 billion in 2022. Forecasts from the State of IoT—Spring 2022 report project indicated that an additional 27 billion connected devices expected to join the internet by 2025 [1]. The proliferation of wireless connections and the diverse distribution of internet-connected devices, ranging from sensors and smartphones to autonomous systems and critical applications, has led to the emergence of numerous cyber threats. [2, 3]. Exploiting wireless connection vulnerabilities can compromise the CIA triad principles, encompassing confidentiality, integrity, and availability. McAfee Labs' survey revealed a staggering 118% surge in ransomware attacks during the first quarter of 2019. Moreover, the use of PowerShell witnessed a significant 460% rise in handling attacks on vulnerable devices [4].

Khaled Bakour or Halit Bakır: Due to the author's dual citizenship, his name can be written in two different ways.

✉ Halit Bakır
halit.bakir@sivas.edu.tr

Özlem Ceviz
ozlemceviz@hacettepe.edu.tr

[1] Department of Computer Engineering, Faculty of Engineering and Natural Sciences, Sivas University of Science and Technology, Sivas, Turkey

[2] WISE Lab, Department of Computer Engineering, Hacettepe University, Ankara, Turkey

Published online: 12 April 2024

Securing both IoT devices and their associated networks is paramount to safeguarding against cyber threats. Organizations allocate substantial funds to enhance security, considering the most significant challenges lie in sectors like healthcare, banking, telecommunications, energy, and government. Numerous cryptographic methods are being suggested to thwart attacks targeting these environments. [5]. The effectiveness of current methods is limited by the dynamic network structure and the heterogeneous distribution of IoT devices. As a result, there is a pressing demand for systems capable of detecting attacks in computer networks, with a specific focus on the IoT environment. These systems are known as intrusion detection systems (IDSs). Intrusion detection systems (IDSs) are designed to monitor, recognize, and assess events within a computer system or local domain, aiming to identify malicious activities. These systems provide a range of options for effectively managing threat and vulnerability risks. [6]. IDS systems can be categorized into three main types: signature based, anomaly based, and specification based. Signature-based IDS compares current traffic patterns with known attack signatures to identify matches, effectively detecting known attack types. However, it may struggle with detecting unknown or zero-day attacks. On the other hand, anomaly-based IDS detects anomalies by comparing them with profiles of normal system behavior, allowing it to identify deviations from expected patterns. In specification-based IDS, deviations from system standards are flagged as potential attacks. This type combines the advantages of both signature-based and anomaly-based IDS, offering a more comprehensive approach to intrusion detection. Intelligent artificial-based anomaly detection systems are widely favored in the literature due to their numerous advantages [7–9]. These methods excel at early detection of new or mutated attacks through models trained with existing samples. However, determining the optimal hyperparameters during model construction and training poses a challenge, as these systems rely on various algorithms. To enhance the intrusion detection performance of AI-based IDS, it is crucial to fine-tune the hyperparameters of the utilized machine learning algorithms and deep learning models [10]. Traditionally, hyperparameter tuning involves manually testing various values for the hyperparameters to assess the model's performance. However, this method is time-consuming and subjective, relying on human observation, leading to questions about its reliability [11]. To address these limitations, automating the hyperparameter optimization process becomes essential, saving time and reducing human effort. This automation can be achieved through different algorithms such as grid search, random search, Bayesian, and genetic algorithms.

In this work, we developed an anomaly-based IDS system by leveraging multiple machine learning models and employing the genetic algorithm for hyperparameter tuning. The CICIDS2017 dataset was chosen for training and testing our model. Initially, the dataset underwent pre-processing, and various machine learning models with their default parameters were used to detect and classify attacks and their variants. Next, we introduced a hybrid feature selection method to identify the most relevant features for the task. The used ML models that performed best in the initial stage were retrained using the selected sub-feature group from the proposed feature selection method. Finally, we optimized the selected models' detection accuracy by employing the genetic algorithm to select the best-performing hyperparameters.

2 Motivation

The growing prevalence of cyber threats has intensified the need for robust and efficient intrusion detection systems (IDSs) to safeguard critical computer networks. In this study, we present a novel and comprehensive approach for developing an anomaly-based IDS system. By harnessing the power of multiple machine learning models and integrating the genetic algorithm for hyperparameter tuning, we aim to significantly improve the system's detection accuracy and adaptability. Our research utilizes the CICIDS2017 dataset, a widely recognized benchmark dataset in the field, for training and testing the proposed model. To address the challenges posed by dynamic network structures and the diverse distribution of Internet of Things (IoT) devices, we meticulously preprocess the dataset. We systematically evaluate various machine learning models, exploring their default parameters to detect and classify attacks and their variants accurately. Recognizing that feature selection plays a crucial role in enhancing performance, we introduce a hybrid feature selection method to identify the most relevant features for our task. Taking the analysis one step further, we retrain the algorithms that exhibit optimal performance in the initial stage, utilizing the selected sub-feature group from our proposed feature selection method. This step ensures that our IDS system is fine-tuned to focus on the most discriminative features, enhancing its precision in detecting and classifying anomalous activities. The second side of our contribution lies in the application of the genetic algorithm for hyperparameter tuning for more polishing of the proposed IDS model's performance. By automating this process, we reduce human intervention and ensure that our IDS system is optimized to deliver the best possible results. The genetic algorithm helps fine-tune the machine learning models' hyperparameters, leading to enhanced accuracy, adaptability, and robustness in the proposed IDS system. Our research is expected to significantly advance the field of intrusion detection by introducing a comprehensive IDS system that outperforms existing solutions. The combination of multiple machine learning models,

hybrid feature selection, and genetic algorithm-based hyperparameter tuning contributes to a versatile and efficient system capable of detecting a wide range of cyber threats, ultimately fortifying the security of critical computer networks and IoT environments.

The rest of the paper is organized as follows: Section 2 illustrates the most important previously conducted works in this domain. Section 3 elaborates on our chosen research methodology, detailing the approaches and techniques employed in our investigation. Section 4 introduces our proposed approach. Following this, in Sect. 5 we delve into the Model Evaluation Metrics employed to assess the performance of our proposed approach. In Sect. 6, we provide insights into the Hardware and Software Platform utilized for our experiments, offering transparency on the computational environment. Subsequently, in Sect. 7 we present our Experimental Results, showcasing the outcomes of our empirical studies. Section 8 is dedicated to a comprehensive Comparison Study. In Sect. 9, we extend our investigation by Testing the proposed method using different datasets, examining its robustness and generalizability. Finally, in Sect. 10 we draw together our findings and insights, culminating in the conclusion, where we summarize the key contributions of our work and discuss avenues for future research.

3 Related Work

3.1 Machine Learning-Based IDSs

A good number of studies propose using artificial intelligence, deep learning, and machine learning approaches for attack detection using different datasets [12, 13]. For example, in [12], the authors apply popular supervised and unsupervised algorithms to detect attacks in the CICIDS2017 dataset. Particularly, a combination of machine learning and deep learning methods has been used to compare the obtained results. The hyperparameters in this work have been tuned manually to choose the values that can give the best results. It has been stated that kNN, DT, and NB algorithms gave higher performance compared with the other used algorithms. In [14], a model based on machine learning algorithm has been proposed. Particularly, the ensemble margin technique has been used to conduct a voting process between multiple algorithms, and the algorithms with the highest votes were selected to provide the highest accuracy. The deep learning methods have been utilized for feature extraction, and SVM and kNN algorithms have been adopted for conducting the classification process. The proposed method has been tested using multiple datasets including UNSW-NB15, CICIDS2017, and NSL-KDD and the results have been compared. Furthermore, the outputs of the kNN and SVM methods are integrated with the Dempster–Shafer classifier method. This integration method has increased the accuracy rate, and 99.84% of success has been achieved in the detection of the U2R attack. In [15], authors focused on machine learning-based IDS using CICIDS2017 dataset. Firstly, three different machine learning approaches namely decision tree, random forest, and SVM are implemented to detect the attacks in the dataset. Then, a machine learning model called the voting classifier (VC) determines the output class with the highest probability between multiple decision. This method improves the detection accuracy to reach 96.25%. Similarly, [16] employed different classification algorithms in order to detect the attacks in the CICIDS2017 dataset and compares their results. It has been stated that the random forest gave the best results between the adopted algorithms. In [17], the paper begins by outlining the importance of IDS in the field of network security and the limitations of traditional IDS techniques. The proposed hybrid algorithm aims to overcome these limitations by improving the accuracy of detection while minimizing false alarms. The authors then explain the principles of tabu search and genetic algorithms and describe how they can be combined to form a hybrid algorithm. The tabu search algorithm is used to explore the search space and generate candidate solutions, while the genetic algorithm is used to optimize these solutions. The proposed IDS system consists of two main components: the training phase and the detection phase. In the training phase, the system learns the normal behavior of the network and creates a profile for each network user. In the detection phase, the system monitors the network traffic in real-time and compares it to the learned profiles to detect any anomalies. The paper presents the results of experiments conducted on the KDD Cup 99 dataset, which is a standard dataset for evaluating IDS systems. The results show that the proposed hybrid algorithm outperforms traditional IDS techniques in terms of detection accuracy and false alarm rate.

3.2 Hyperparameters Tuning for IDSs

The use of artificial intelligence methods in attack detection is increasing due to many features such as scalability, computational ability, and increasingly accurate detections. However, these methods have a large number of parameters, and it is important to increase the accuracy rate by optimizing these parameters. As a result, hyperparameter tuning research has gained popularity in the literature [5–12]. Different types of hyperparameter tuning approaches such as grid search, random search, and Bayesian have been adopted in the literature.

In [18], the goal of the study is to find the best accuracy for network attack detection by fine-tuning various LSTM hyperparameters including optimizers, loss functions, learning rates, and activation functions and comparing their

performance on the CICIDS2017 dataset. The best accuracy obtained after the hyperparameter tuning process was 99.54%. Similarly, [19] demonstrated that tuning hyperparameters had an impact on the performance of machine learning algorithms. Some hyperparameters such as learning rate, iteration numbers, and optimizers have been tuned to select the best values for each of which. The study focused on DDoS attacks and employed simple neural networks and LSTM algorithms for detecting them. The simple neural network obtained an accuracy of 100% when it has been trained using CAIDA and DARPA datasets.

In [20], author proposed machine learning approaches for DDoS attack detection based on CICDDoS2019, which contains 12 different DDoS attacks. Firstly, data pre-processing is performed, which includes organizing, cleaning, and scaling the data samples. Second, a hybrid feature selection method is presented for extracting the best features. Next, grid search was used to tune the hyperparameter by selecting the best parameters to improve detection performance, and the model was trained using the selected best features. It has been stated that the GB model obtained the highest accuracy of 99.97% among all the applied algorithms. In [21], NSL-KDD and CICIDS2017 datasets have been used for training a proposed neural network IDS system. Authors focused on the effect of hyperparameter tuning on the model accuracy. Grid search algorithm has been used for tuning hyperparameters including the number of hidden layers, number of neurons, the activation function, the optimizer, the batch size, and the number of epochs. The study has been conducted to show with experimental results that the smallest change in hyperparameters affects the accuracy of machine learning models. After conducting the hyperparameterd optimization, the accuracy value with the best hyperparameters reached 99%. Similarly, machine learning-based DoS and DDoS attacks detection was carried out using various datasets including ISCXIDS2012, CICIDS2017, CSE-CIC-IDS2018, and CICDDoS2019 in [22]. The hyperparameters of various algorithms were optimized by the grid search algorithm. After parameter optimizations, the RF and DT algorithms obtained an accuracy value of over 98% for all four datasets.

In [11], hyperparameter tuning method has been proposed as a combination of grid search and random search approaches to improve the performance of deep learning model classification performance. Data preparation and preprocessing operations were performed on the NSL-KDD and CSE-CIC-IDS2018 datasets. Grid search was conducted through all hyperparameters to find the best value for each of which. In order to reduce the time of the grid search algorithm, hyperparameter setting was combined with random search, and the process was carried out without a specific order or criteria. In [23], Bayesian optimization algorithm has been adopted for hyperparameter tuning. The authors used an unsupervised learning algorithm for feature extraction, while they used deep learning techniques for intrusion detection. Activation function and weight hyperparameters were tuned to increase the performance of the adopted deep learning model. As a result of the evaluation using the BoT-IoT dataset, the accuracy value increased to 99.99% thanks to the hyperparameter setting.

In [24], the paper addresses the need for effective network intrusion detection systems for cloud IoT devices. It highlights the use of CNN architecture and transfer learning. In this case, the knowledge learned from a base dataset is transferred to the IDS on cloud IoT devices. Five CNN model trained and two datasets are used: CICIDS2017 and CSE-CICIDS2018. It discusses the accuracy, precision, recall, and F1 score achieved by the system. After hyperparameter tuning, accuracy was obtained at 0.9999 for both datasets. In [25], authors emphasize the importance of anomaly-based detection, where abnormal patterns or behaviors are identified as potential attacks. The model is trained using a deep neural network (DNN), and tuning hyperparameters and a filter-based feature selection approach are used to get the highest performance on the UNSW-NB15 dataset. Without data balancing, the proposed model has an accuracy of 84%. The final score after data balancing is 91%. Similarly, in [26], the paper presents a novel approach for intrusion detection in IoT environments by combining deep reinforcement learning, feature selection, and optimal hyperparameters. The proposed system combines filter-based, wrapper-based, and embedded feature selection methods. Feature selection methods are applied to the NLS-KDD dataset to select related features. The optimal hyperparameters for the deep reinforcement learning (DRL) algorithm are determined using a swarm-based metaheuristic optimization algorithm called the whale optimization algorithm (WOA). Proposed method increases accuracy with feature selection methods. In addition, selecting the appropriate hyperparameter is critical to the efficiency of IDS performance.

In [27], attack detection performance was evaluated with stacked LSTM and bi-directional LSTM techniques applied to UNSW-NB15 and BoT-IoT datasets. With hyperparameter optimization, the LSTM method gives an accuracy of 96.60% in the UNSW-NB15 dataset, while the result for the BoT-IoT dataset is 99.99%. Bi-directional LSTM gives 96.41% and 99.99% accuracy values for these datasets, respectively. In a recent paper [28], authors propose a model for detecting attacks in BoT-IoT dataset. The proposed model uses a kNN classifier and feature selection techniques to identify and classify network intrusions. The authors claim that their model is more efficient and accurate than existing models. In addition, to enhance data quality and choose the best-performing features, the principal component analysis (PCA), univariate statistical test, and genetic algorithm (GA) are utilized for feature selection. With the GridSearchCV

Table 1 Brief information about some previously conducted works

References	Year	Methods	Hyperparameter tuning algorithm	Datasets
[12]	2021	ANN, DT, k–NN, NB, RF CNN, SVM, EM, k–means SOM	–	CICIDS2017
[14]	2021	SVM, k–NN	–	UNSW–NB15 CICIDS2017 NSL–KDD
[15]	2022	DT, RF, SVM	–	CICIDS2017
[16]	2020	CNN, NB, RF	–	CICIDS2017
[31]	2020	LSTM	Manually	CICIDS2017
[19]	2019	Basic Neural Network	Manually	CAIDA and DARPA
[32]	2021	RF, DT, DT, RF	Gridsearch	– ISCXIDS2012 – CICIDS2017 – CSE–CIC–IDS2018 – CICDDoS2019
[20]	2021	GB	Gridsearch	CICDDoS2019
[11]	2021	DNN	Gridsearch, Random search	CSECIC–IDS2018
[23]	2020	DNN	Bayesian	BoT–IoT
[24]	2023	CNN	BO–TPE	– CICIDS2017 – CSE–CICIDS2018
[25]	2023	DNN	–	UNSW–NB15
[26]	2022	Deep reinforcement learning	–	NLS–KDD
[27]	2022	LSTM Bi–Directional LSTM	–	UNSW–NB15 Bot–IoT
[28]	2023	kNN	Grid search	Bot–IoT
[29]	2023	LGBM	GA	DS2OS
[30]	2022	SVM RF	–	BoT-IoT

hyperparameter tuning method, the best parameters of the kNN algorithm were selected, and the optimum result was found. The authors propose an advanced and optimized light gradient boosting machine (LGBM) technique to identify intrusive activities in the Internet of Things (IoT) network in [29]. The dataset utilized in this study was initially designed for anomaly detection in IoT service accesses and is known as the distributed smart space orchestration system (DS2OS). The proposed approach is used genetic algorithm (GA) to optimize the hyperparameters. The paper aims to improve the accuracy of intrusion detection in IoT networks by using the proposed model. The paper compares the proposed model with other machine learning techniques such as support vector machine (SVM), random forest (RF), and decision tree (DT). The results show that the proposed model outperforms other machine learning techniques in terms of accuracy, precision, recall, and F1 score. In [30], The authors suggest that traditional intrusion detection systems (IDSs) are not effective in detecting attacks in IoT networks due to the unique characteristics of these networks, such as the large number of devices, the heterogeneity of devices, and the limited resources of devices. The proposed framework consists of anomaly detection and multi-class classification. The authors use UNSW BoT-IoT dataset to evaluate the performance of the proposed framework. Random forest algorithm is used for multi-class classification, and the SVM algorithm is applied for anomaly detection. Hyperparameters are tuned for both scenarios. The results show that the proposed framework outperforms traditional IDS in terms of accuracy, precision, and recall. The Table 1 illustrates a brief information about some important works conducted previously in this domain.

Our proposed approach stands out from surveyed methodologies by integrating a robust optimization technique—the genetic algorithm. This method is particularly valuable for

addressing complex, nonlinear relationships within the algorithm. Mirroring the process of natural selection, the genetic algorithm ensures the survival of the fittest set of parameters, thereby optimizing the algorithm for the specific task at hand. This sophisticated optimization sets our approach apart, providing a more nuanced and effective solution compared to conventional tuning methods. The result is the identification of optimal parameters, enhancing the adaptability and overall performance of our approach. Additionally, our approach introduces a novel hybrid feature selection method, a critical factor in improving the accuracy and detection time of intrusion detection systems. This method strategically combines various feature selection techniques, leveraging their respective strengths to identify and retain the most relevant features for the detection process. This contributes not only to increased detection accuracy and reduced overhead time but also ensures the model's robustness by focusing on the most influential features within the dataset. By synergizing the power of a genetic algorithm for algorithmic tuning with the effectiveness of a hybrid feature selection method, our proposed approach offers a comprehensive and innovative solution that surpasses conventional methods found in the literature.

4 Research Method

4.1 Machine learning Algorithms

Artificial Intelligence (AI) is a broad field encompassing various technologies and methodologies aimed at endowing machines with human-like intelligence. Within AI, machine learning (ML) and deep learning (DL) stand out as key subfields. Machine learning involves algorithms and statistical models that enable systems to improve their performance on a specific task over time, learning from data without explicit programming. Deep learning, a subset of machine learning, focuses on neural networks with multiple layers (deep neural networks) to simulate the intricate processing patterns of the human brain. These technologies find applications across diverse domains, including natural language processing, computer vision, speech recognition, cyber security, and autonomous systems [33–45], showcasing their versatility and impact on various facets of modern technology.

4.1.1 Decision Tree

A decision tree (DT) was introduced by Quinlan [46] as a supervised learning algorithm that expresses groupings and divisions using a binary tree flow chart. DT typically begins with a single node before branching out to other possibilities results. Each of these results generates new nodes that branch out into other instances [47]. As a result, in order to classify a data point, one should start at the decision tree's root and work their way up the tree until they reach the leaf node, which is represented by the branch that each test's result indicates. The resultant classification is given by the name of the class at the leaf node.

4.1.2 Random Forest (RF)

Random forest (RF) [48], developed by Breiman, aims to provide low classification error by creating a large number of decision trees from randomly selected data [49]. Since they make the same sorts of prediction errors, randomly produced trees are less related and could reduce overfitting the model. Each tree created gives a vote for the classification, and the one with the most votes is determined to be the final prediction.

4.1.3 Naïve Bayes (NB)

NB is a method based on Bayes' theorem and using probabilistic classifiers. This method assumes that each feature is independent of the others. Without taking advantage of the interactions and relationships between features that are important in distinguishing between classes, this method may not perform well in complex tasks. However, it has advantages in terms of ease of use, simplicity, and the ability to work with low training examples [9].

4.1.4 XGBoots

Presented by the Distributed Machine Learning Community (DMLC) XGBoost was primarily created utilizing gradient-boosted decision trees for speed and performance [50]. XGBoost first generates ordered decision trees, after which all data are chosen by assigning a weight value that is initially constant but varies depending on the analysis [51]. This algorithm has a high tolerance for missing values, and classification is done by strengthening an already trained model with new data. It is an effective method for optimum use of resources and reduction of computation time.

4.1.5 Stochastic Gradient Descent Classifier (SGD)

The stochastic gradient descent (SGD) [52] method, which is effective and simple, employs approximate gradients computed from subsets of the training data and updates the parameters in real time. With its ability to handle large datasets and handle training instances individually, SGD classifier provides several benefits.

4.2 Optimization Methods and Techniques

4.2.1 Feature selection

Feature selection is the process of identifying the most relevant features or variables that can predict the target variable in a dataset. There are multiple algorithms can be used for conducting feature selection. In this work, we chosen to use two feature selection approaches in hybridized manner each of these approaches will be described briefly in this section:

Mutual Information-based Feature Selection (MIFS): MIFS [53] is a feature selection method that is based on the mutual information measure between the input features and the target variable in a dataset. It selects a subset of features that maximize the mutual information with the target variable while minimizing the mutual information between the selected features. The MIFS algorithm works in a sequential manner, where it starts by selecting the feature with the highest mutual information with the target variable. In each subsequent step, it selects the feature that has the highest mutual information with the target variable, conditioned on the previously selected features. The algorithm stops when a predefined number of features have been selected or when the mutual information between the selected features is above a certain threshold.

Sequential Feature Selection (SFS): SFS [54] is a type of wrapper feature selection method that selects the best subset of input features by evaluating the performance of a machine learning algorithm on each subset of features. SFS works by iteratively adding or removing features from the current subset until the best-performing subset is found. The SFS algorithm starts with an empty subset of features and adds one feature at each time based on some selection criterion, such as maximum accuracy or minimum error. At each step, the algorithm evaluates the performance of the machine learning algorithm on the subset of features and selects the feature that improves the performance as possible. The algorithm continues until the predefined number of features or some stopping criterion is met. SFS can be performed in a forward or backward manner. Forward selection starts with an empty subset and adds features one at a time, while backward selection starts with the full set of features and removes one feature at a time. Backward selection can be more computationally efficient as it avoids evaluating all possible feature subsets.

4.2.2 Hyper Parameter Tuning GA

The genetic algorithm [55] was developed and introduced in 1960 by John Holland, University of Michigan students, and colleagues. This algorithm is based on inspiring by natural evolutionary processes [56]. Particularly, this algorithm simulates natural selection, known in evolutionary theory as survival of the fittest. For the solution of an optimization problem, the suitability of each parameter set is checked, and the most suitable parameters are tried to be determined. The search space is represented as a grouping of individuals known as chromosomes. Gene refers to the set of characteristics that identify an individual. In order to select the most suitable parameters, the goodness of each chromosome [57] should be evaluated with a "Fitness Function". Mutation, crossover, and selection processes are used in the evaluation process to ensure natural selection. The best individuals are chosen to progress through crossover, mutation, or selection [58] until a new population is formed. As a result, the optimization problem's solution is determined to be the population's fittest or best members, who are then identified.

4.3 Dataset

The performance of the model is evaluated using the CICIDS2017 [59] dataset published by the Canadian Institute. The dataset was collected using simulation environment over the course of five days, including attacks and normal traffic scenarios, which produced data that was very close to reality. This dataset covers the abstract characteristic attitudes of 25 users in accordance with the HTTP, HTTPS, FTP, SSH, and email protocols. According to the 2016 McAfee Report, the dataset consists of a variety of different attacks, including brute force FTP, brute force SSH, DoS, Heartbleed, web, infiltration, botnet, and DDoS attacks, which were not present in any of the previous datasets [12]. Each data sample in the dataset has about 80 features. Although the dataset has a distinct advantage, it does have some drawbacks such as including NaN values and distribution of attacks on eight CSV files [15]. However, these disadvantages can be solved by pre-processing the dataset. Figure 1 shows information about the classes of the original dataset.

5 Proposed Approach

The proposed method comprises four major phases: pre-processing, feature selection, classification, and hyperparameter tuning. The first phase is pre-processing phase which will be described in the next section.

5.1 Pre-Processing

It is a crucial phase in the improvement of any machine learning framework and is mostly used to organize and clean raw data in order to make sure that it is suitable for the creation and training of any machine learning model. In the pre-processing phase, we performed exploratory data analysis to understand the dataset distribution using visualization techniques. Subsequently, data cleaning and feature scaling were applied to normalize the range of features. Categorical

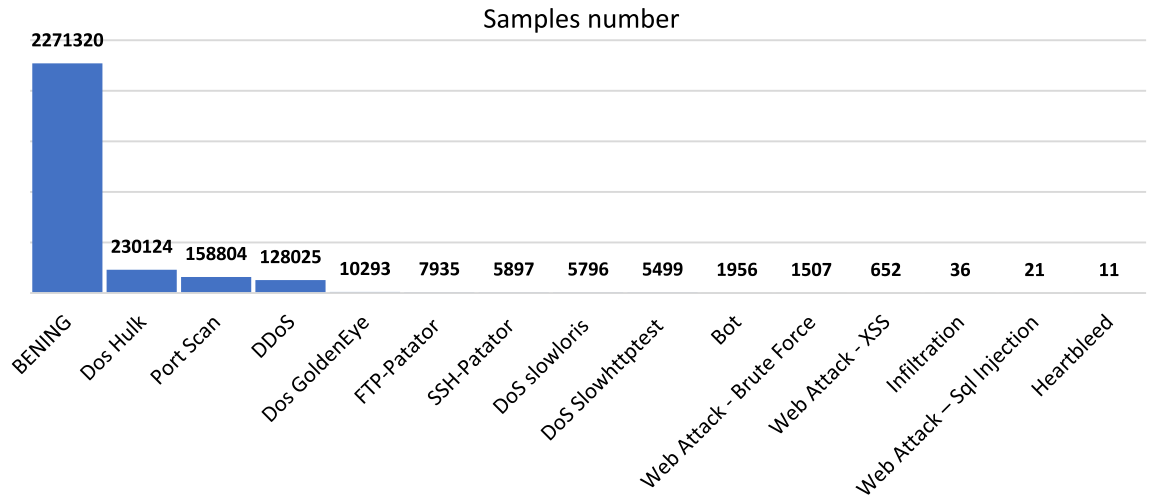

Fig. 1 The data distribution in the classes of the CICIDS2017 dataset

data were converted into numerical data using label encoding. This critical pre-processing phase significantly enhances classification results. The following are the pre-processing steps that have been conducted in this work:

5.1.1 EDA

Exploratory data analysis (EDA) is conducted to investigate, summarize, and visualize the data distribution in the dataset. This step aids researchers in comprehending and interpreting the dataset through various techniques. EDA provides valuable insights into data types, category distributions, the presence of NaN data and duplicates, and the identification of outlier points that require cleaning. Additionally, correlation values between features are observed after completing the data analysis.

5.1.2 Data Cleaning

Following the completion of EDA, the data cleaning process is initiated to prepare the dataset for training machine learning models effectively. Empty and infinite values, along with duplicate rows, are removed to enhance model performance and reduce computation time. Additionally, certain attacks (classes) with limited samples and minimal impact on model performance are excluded from the dataset. The final dataset includes Brute Force, DoS, PortScan, and DDoS attacks.

5.1.3 Handling Imbalanced Data

In this work, the attack detection process was carried out in two distinct scenarios. In the first scenario, the dataset's data were classified as either attack or benign, representing a binary classification task. In the second scenario, the data were categorized as benign or attack, and the detected attacks were further classified into four different attack families, forming a multi-class classification problem. So, we handled the imbalanced data in two different scenarios based on the type of classification process to be conducted:

Binary Classification Initially, we merged the attack data with different labels in the dataset (Brute Force, DoS, PortScan, and DDoS attacks) and assigned them a common "attack" label, thus transforming the dataset into a binary classification format. Subsequently, we observed that the number of "Benign" data points exceeded the number of "attack" data points. To address this class imbalance and improve performance, we employed the SMOTE algorithm. By utilizing this technique, we increased the number of samples in the "attack" class from 421,603 to 2,072,444, thereby equalizing its data samples with the "Benign" class.

Multi-Class Classification Following pre-processing, the dataset consists of 2,072,444 benign samples and 421,603 attack samples, distributed among four different classes. To address the class imbalance, we developed a hybrid data-balancing technique, comprising both downsampling and upsampling processes. Initially, we performed downsampling to equalize the number of benign data samples with that of the attack data classes, resulting in 421,603 benign samples. Subsequently, SMOTE was applied to increase the data samples in each attack class to reach 421,603 samples each. This approach effectively balanced the dataset, enabling more reliable and accurate model training.

5.1.4 Feature Scaling and Label Encoding

When training a model with features of varying scales, the process can become complex, time-consuming, and may lead

to occasional model failures [20]. To circumvent these challenges, scaling techniques are employed. In this study, we utilized the StandardScaler, which standardizes the values of numerical columns in the dataset while preserving the variances in the value ranges. The StandardScaler is represented using the following question:

$$z = \frac{x - \mu}{\sigma}$$

where z is the standardized value. x is the original value of the feature. μ is the mean of the feature. σ is the standard deviation of the feature.

Additionally, since the CICIDS2017 dataset contains multi-class data with non-numeric values, we associated these values numerically using the Label encoder. Each class was assigned a numeric value starting from 0, and the machine learning algorithms leveraged these numerical representations for performing multi-class classification. This approach ensures compatibility and efficiency in the model training process.

5.2 Proposed Model Optimization Method

After pre-processing the dataset, we utilized our proposed data-balancing technique to achieve an even distribution of samples across classes. This process led to the creation of two datasets: one with two classes (benign and malicious) and another with five classes (one benign and four types of attacks), allowing the application of our method to both binary and multi-class classification. Following this, we devised a multi-stage experimental process to pinpoint the optimal machine learning algorithm for enhancing performance in attack detection. Initially, we evaluated machine learning algorithms with their default hyperparameters using the balanced dataset. From this assessment, we identified the top three algorithms for subsequent stages. In the following stage, we introduced a hybrid feature selection method to optimize the dataset, enhancing the performance of machine learning algorithms in attack detection. The feature selection process involved a novel hybrid approach combining mutual information, a well-established feature selection method, with sequential feature selection. Initially, mutual information was employed to identify the best 35 features from the original 80. Subsequently, sequential forward feature selection further refined the feature set, ensuring the most effective features for accurate attack detection. This feature selection method was applied to both binary and multi-class classification datasets. For the binary classification dataset, the proposed feature selection yielded six selected features: Destination Port, Flow Duration, Bwd Packet Length Max, Flow Bytes/s, Bwd IAT Std, and Bwd URG Flags. Meanwhile, the multi-class classification dataset featured six selected features: Destination Port, Flow Bytes/s, Flow IAT Mean, Fwd IAT Std, Bwd IAT Max, and Bwd IAT Min. Subsequently, the top three machine learning algorithms, identified in the previous stage, were retested using the reduced feature dataset for both binary and multi-class datasets. This process led to the selection of two algorithms—RF and XGBoost—that demonstrated the best results. In the final stage, a hyperparameter tuning process was proposed for the selected algorithms, RF and XGBoost, to further refine their performance. Employing a genetic algorithm during hyperparameter tuning optimized the algorithms, resulting in enhanced performance. Experimental results underscore the varied outcomes of different machine learning algorithms with default parameters when hybrid feature selection methods are employed. Furthermore, the effectiveness of our approach is evident in the improved results achieved after hyperparameter optimization. The methodology is visually summarized in the flow diagram (Fig. 2), and a detailed analysis of our findings is presented in the subsequent sections.

6 Model Evaluation Metrics

In this section, we provide a comprehensive overview of the machine learning metrics employed in the study to assess the performance and efficacy of the proposed model. These metrics play a crucial role in quantifying the model's ability to generalize and make accurate predictions.

6.1 False Positive (FP), False Negative (FN), True Positive (TP), True Negative (TN)

These metrics provide insights into the specific types of errors made by the model.

FP: The number of instances predicted as positive but is actually negative.

FN: The number of instances predicted as negative but is actually positive.

TP: The number of instances predicted as positive is actually positive.

TN: The number of instances predicted as negative is actually negative.

6.2 Accuracy (ACC)

Accuracy represents the ratio of correctly predicted instances to the total instances in the dataset. Accuracy provides an overall measure of the model's correctness in predictions.

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

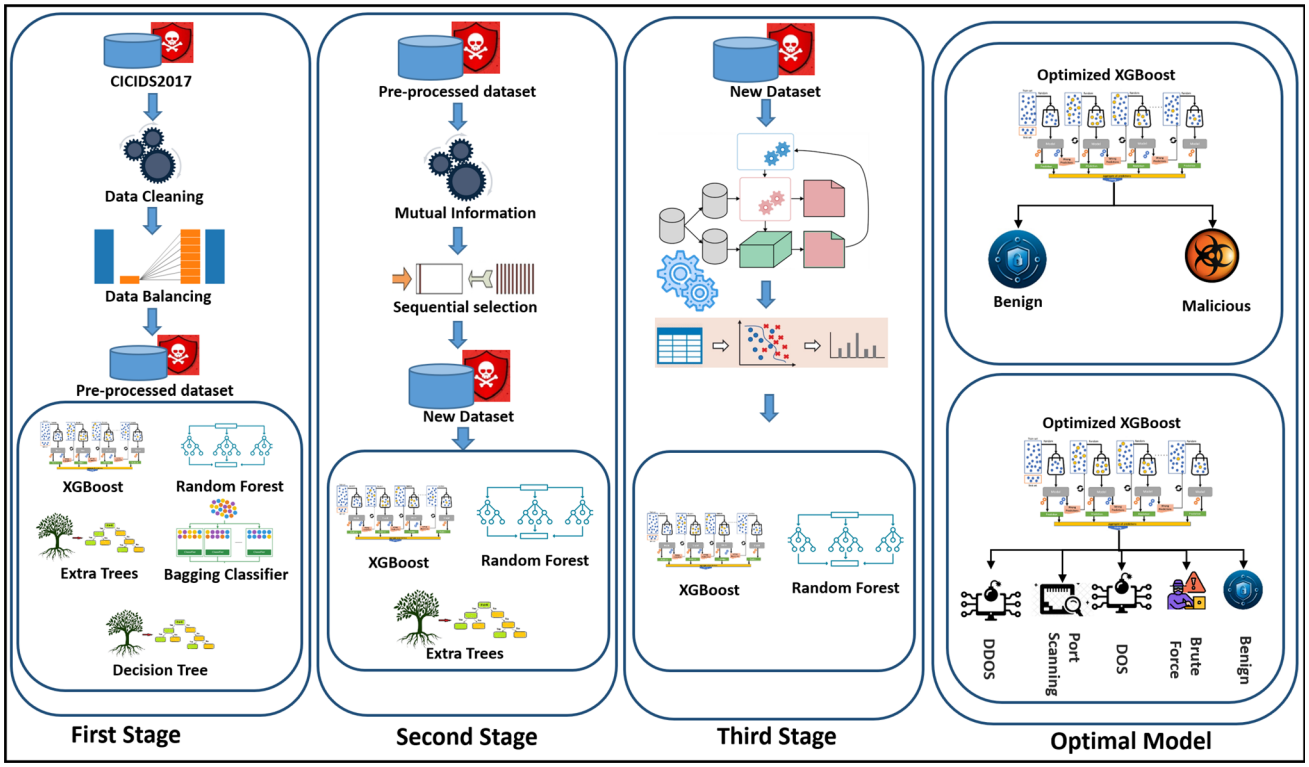

Fig. 2 Proposed model

6.3 Precision

Precision is the ratio of correctly predicted positive observations to the total predicted positives. Precision focuses on the accuracy of positive predictions.

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

6.4 Recall (Sensitivity)

Recall is the ratio of correctly predicted positive observations to the all observations in the actual class. Recall emphasizes the model's ability to capture all relevant instances of the positive class.

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

6.5 F1 Score

F1 score is the harmonic mean of precision and recall, providing a balanced measure between the two. F1 score is particularly useful in imbalanced datasets, where the class distribution is skewed.

$$F1 - \text{Score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

6.6 Confusion Matrix

A confusion matrix is a tabular representation that summarizes the performance of a classification algorithm. It compares the predicted classes against the actual classes and is especially useful for evaluating the performance of a model on a dataset with known class labels.

7 Hardware and Software Platform

The proposed model was implemented using Python, with various libraries, including Scikit-learn, Optuna, skopt, and others, employed throughout the research. Experiments were conducted on a computer with an 13th Gen Intel (R) Core (TM) i9-13980HX 2.20 GHz processor, 32 GB RAM, NVIDIA DeForce RTX 4090 16GB GPU and Windows 11 operating system.

8 Experimental Results

Initially, the CICIDS2017 dataset underwent pre-processing to enhance its suitability for training machine learning algorithms. Subsequently, a data-balancing technique was employed to address class imbalance. Five distinct machine learning classification algorithms were applied using default hyperparameter values, and results were obtained for both binary and multi-classification scenarios. From this initial exploration, the top three performing machine learning algorithms were identified. Following the algorithm selection, a hybrid feature selection approach was implemented on the dataset. The previously chosen three machine learning algorithms were then retrained and tested to assess their detection performance. From this phase, the two algorithms demonstrating the highest accuracy values were chosen for further refinement through hyperparameter tuning. The hyperparameter tuning process utilized the genetic algorithm, leveraging the Optuna library. This algorithmic approach incorporates mutation, crossover, and selection processes to iteratively discover the optimal algorithm and parameters. The results obtained from hyperparameter tuning were compared based on detection accuracy, F1 score, and computational time, providing insights into the impact of this tuning process on the intrusion detection system's performance. Given that the research covered both binary classification and multi-class classification scenarios, the outcomes were categorized accordingly. The experimental findings and methodologies employed in this study are visually represented in Fig. 3.

We initially explored five machine learning algorithms—XGBoost, random forest, decision tree, bagging, and extra tree algorithms. Through the evaluation process, we identified the most optimal classifier capable of achieving the task with superior performance. We enhanced the intrusion detection system's detection performance through multiple stages. In the following sections, we will describe the results obtained from the multi-class classification and binary classification experiments in details separately.

8.1 Multi-Class Classification Results

In the first scenario, we proposed classifying attacks in the dataset into their respective families. Initially, we addressed the dataset's imbalance using our hybridized balancing approach. Subsequently, our proposed three stages of optimization were applied. The first stage involved training five different machine learning algorithms on the balanced dataset, from which the top three algorithms were selected. In the second stage, we applied the hybridized feature selection method, refining the dataset's best features. The chosen three machine learning algorithms were then tested after the feature selection process, and the top two algorithms were selected. Finally, we employed the genetic algorithm to tune hyperparameters for the chosen two algorithms. The optimized hyperparameter values were utilized, and these two

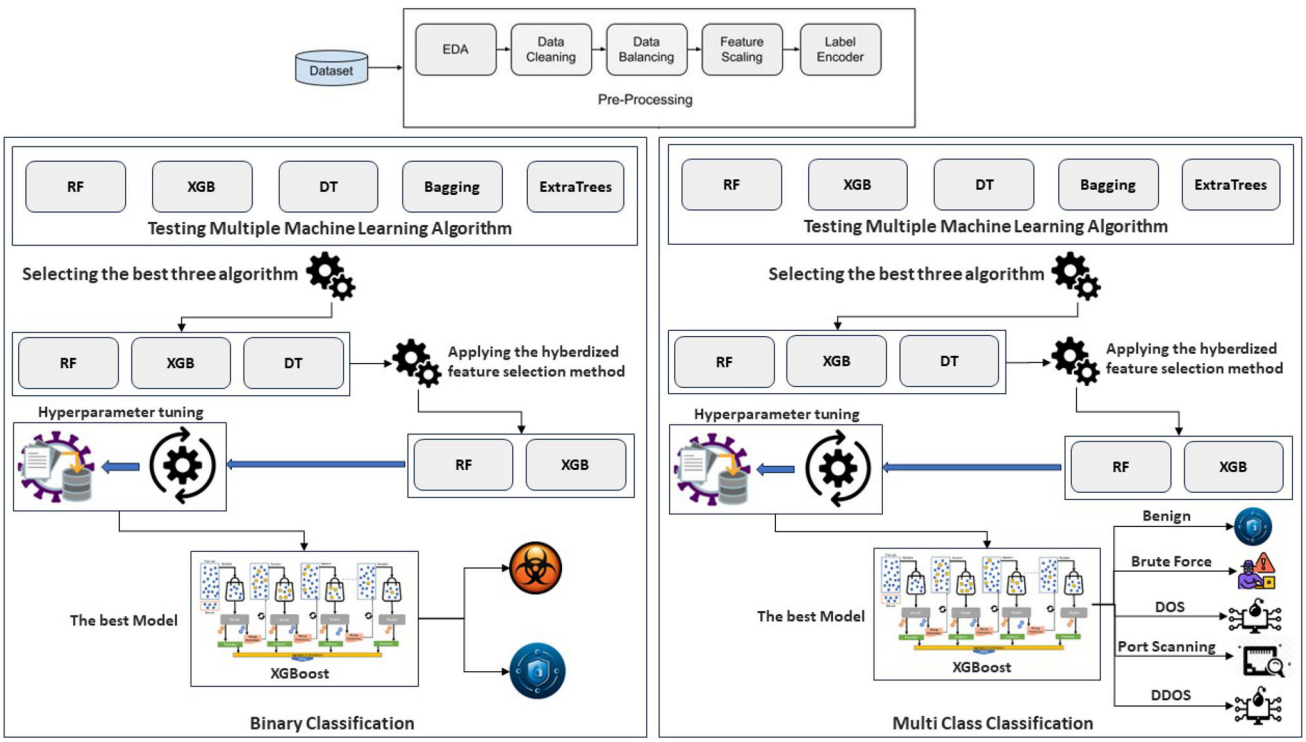

Fig. 3 The Experimental Framework

algorithms were trained and tested to select the best optimized model. Results are presented under three subtitles: first stage results, second stage results, and third stage results.

8.1.1 First Stage Results

In the first stage, we trained five ML algorithms—XGBoost, random forest, decision tree, bagging, and extra tree—using their default hyperparameters. The dataset underwent hybrid data balancing before training. Table 2 illustrates the obtained results. Notably, the XGBoost classifier demonstrated superior performance with a detection accuracy and F1 score of 99.98%. Despite the extra tree algorithm's comparatively lower performance, it still achieved a high detection accuracy of 99.96% and an F1 score of 99.96%. Additionally, the XGBoost algorithm outperformed others in computational time, completing the task in 70.15 seconds. We proceeded to the next step with XGBoost, RF, and bagging algorithms.

8.1.2 Second Stage Results

In this stage, our proposed hybrid feature selection approach was applied to identify the optimal features from the dataset. Subsequently, the three ML algorithms were trained using these selected features to optimize both detection accuracy and training time. The multi-class classification dataset featured six selected features: Destination Port, Flow Bytes/s, Flow IAT Mean, Fwd IAT Std, Bwd IAT Max, and Bwd IAT Min. The results, showcased in Table 3, reveal a slight decrease in detection accuracy across all algorithms. However, this is deemed acceptable given the significant reduction in computational time for all ML algorithms. Notably, XGBoost maintained its superior performance, achieving 99.95% accuracy and an F1 score of 99.95%. Furthermore, the computational time of XGBoost notably decreased from 70 seconds to 23 seconds. At the end of this stage, we selected the best two ML algorithms in this scenario XGBoost and RF algorithms.

8.1.3 Third Stage Results

We conducted hyperparameter tuning to identify optimal parameters for both the RF and XGBoost algorithms, aiming to achieve the highest accuracy values. Table 4 outlines the default parameter values of these algorithms prior to optimization and their respective values after the tuning process. Subsequently, applying the optimized parameters for attack detection, the RF algorithm demonstrated an accuracy of 99.96%, and the XGBoost algorithm achieved 99.97% accuracy. So, there are no notable changes in the performance of the proposed ML algorithms after the hyperparameter tuning process and feature selection process. Yet, notably, we observed that comparable performance to the machine learning algorithms with their default hyperparameters can be achieved with a reduced computational time. While the XGBoost algorithm initially required 70 s, it now completes the task in approximately 42 s. In contrast, we observed a substantial reduction in the processing time for the random forest classifier. Initially requiring 891 s, it now completes the task in just 6.5 s following the feature selection and hyperparameter tuning processes. Table 5 illustrates the results obtained in this stage. Furthermore, Fig. 4 illustrates the confusion matrixes for the optimized models.

Table 3 Second stage optimization results for multi-class classification

Algorithm	Accuracy	F1 score	Time (Second)
XGBClassifier	0.999504	0.9995	23.56194353
RandomForestClassifier	0.999464	0.9995	17.99695563
BaggingClassifier	0.99932	0.9993	4.260981798

8.2 Binary Classification Results

In the second scenario, we proposed aggregating all attack classes in the dataset into a single-class labeled 'attacks'. To tackle data imbalance, we employed the SMOTE oversampling algorithm, which increased the number of attack data samples. Subsequently, our three-stage optimization process was applied to the modified dataset. In the initial stage, we conducted binary classification using five machine learning algorithms with their default hyperparameter values. The top three performing algorithms were then selected for the subsequent stage. In the second stage, our hybridized feature selection method was applied to identify the most effective

Table 2 First stage optimization results for multi-class classification

Algorithm	Accuracy	F1 score	Time
XGBClassifier	0.99977420	0.99980000	70.14908648
RandomForestClassifier	0.99971347	0.99970000	891.35473275
BaggingClassifier	0.99966414	0.99970000	916.54004097
DecisionTreeClassifier	0.99960721	0.99960000	120.90044785
ExtraTreesClassifier	0.99958255	0.99960000	303.94238830

Table 4 Best hyperparameters of XGBoost and RF chosen by GA

Algorithms	Parameters	Default	Best values
XGBoost	max_depth	6	9
	learning_rate	0.3	0.0399
	n_ estimators	100	496
	min_child_weight	1	4
	gamma	0	0.6290
	subsample	1	0.9858
	colsample_bytree	1	0.6961
	reg_alpha	1	1.7520054800488898e-06
	reg_lambda	1	0.1845
RF	max_depth	None	30
	min_samples_leaf	1	2
	min_samples_split	2	10
	n_estimators	100	200

Table 5 Third stage optimization results for multi-class classification

Algorithm	Accuracy	F1 score	Time (Second)
XGBClassifier	0.9997	0.9998	42.016361951828
RandomForestClassifier	0.9996	0.99954	6.564135789871216

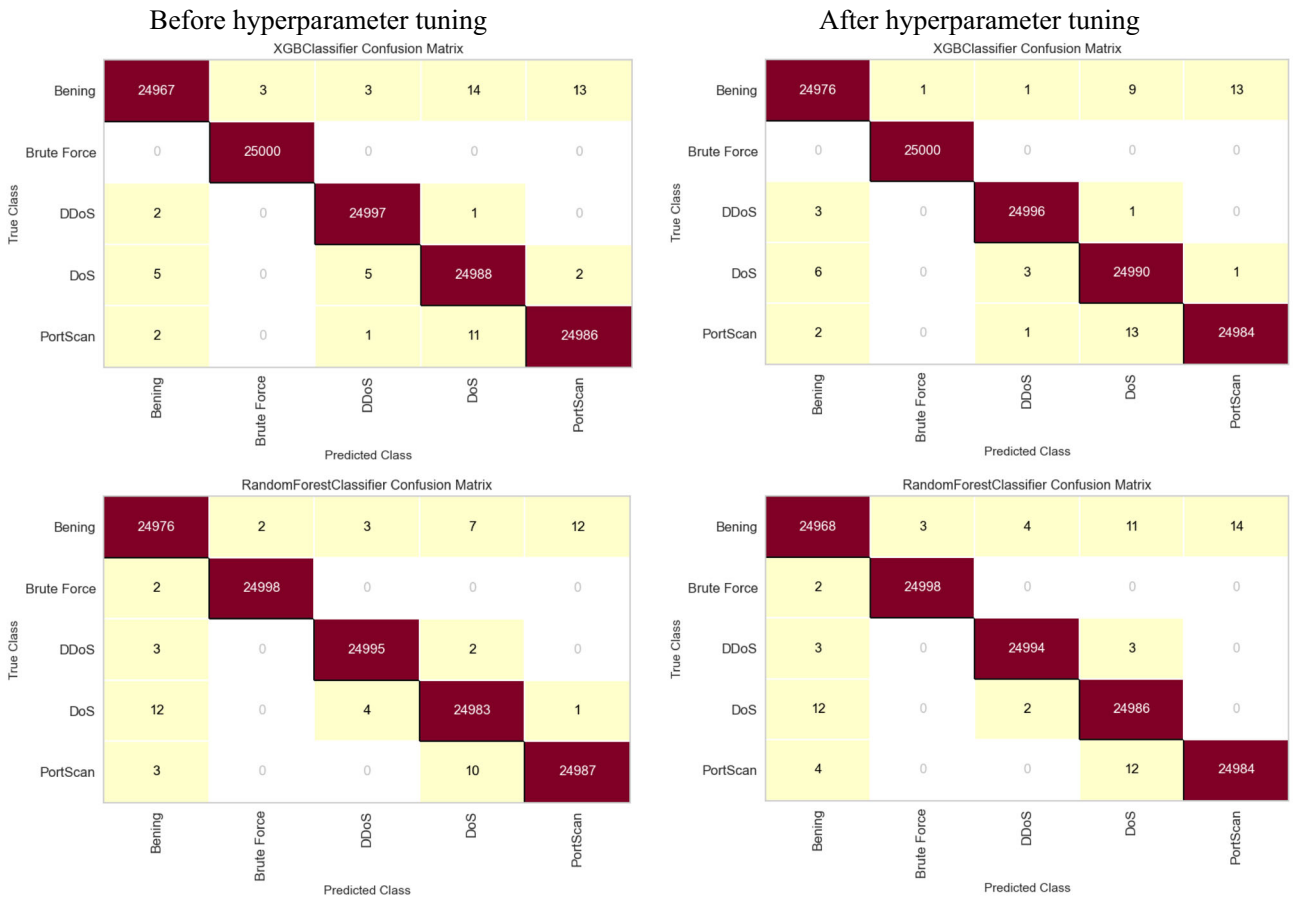

Fig. 4 The confusion matrices of xgboots and RF algorithms after the hyperparameter optimization for Multi-Class Classification

Table 6 First stage optimization results for binary-class classification

Algorithm	Accuracy	F1 score	Time (Second)
XGBClassifier	0.999629423	0.9996	32.94350481
RandomForestClassifier	0.999533884	1	1470.253617
BaggingClassifier	0.999390092	0.9994	4828.733421
DecisionTreeClassifier	0.999193223	0.9992	282.3575733
ExtraTreesClassifier	0.999118915	0.9991	519.2035694

feature group for optimal results with the machine learning algorithms. Within this stage, the best two ML algorithms were chosen to advance to the final stage. In the last stage, the genetic algorithm was employed to optimize the performance of the two selected machine learning algorithms. Ultimately, the best ML algorithm was chosen as an optimized IDS system capable of detecting potential attacks with high performance and minimal computational time. Further discussion of the case study findings will be presented in subsequent subsections.

8.2.1 First Stage Results

Initially, we utilized the XGBoost, random forest, bagging, decision tree, and extra tree machine learning algorithms with their default hyperparameters, training them on the balanced dataset. Table 6 showcases the results, indicating accuracies of 99.96%, 99.95%, 99.94%, 99.92%, and 99.91%, respectively. Similar to the multi-classification scenario, the XGBoost algorithm exhibited the highest accuracy and the lowest false-negative value for binary classification. Additionally, it was noted that the XGBoost algorithm achieved the task with the shortest computational time. At this stage's conclusion, we selected the top three ML algorithms for the subsequent stage: XGBClassifier, RandomForestClassifier, and BaggingClassifier.

8.2.2 Second Stage Results

Finally, the hybrid feature selection method was applied to the balanced dataset. Initially, 35 features were selected using mutual information, and the sequential feature selector was then employed to further narrow down the selection to just six features. For the binary classification dataset, the proposed feature selection yielded the following selected features: Destination Port, Flow Duration, Bwd Packet Length Max, Flow Bytes/s, Bwd IAT Std, and Bwd URG Flags. After determining the optimal features, the new dataset was tested using the default parameters of the XGBClassifier, RandomForestClassifier, and BaggingClassifier algorithms. In this evaluation, XGBoost achieved an accuracy of 99.88%, RF achieved 99.88%, and Bagging achieved 99.86%. Thus, by selecting only six features, we achieved nearly the same performance using the XGBoost classifier with only 3.36 s, compared to 32.94 s in the first stage. As a result of this stage, we selected the best two ML algorithms, namely XGBoost and random forest, to be used in the final stage. The results obtained in this stage illustrated in Table 7.

Table 7 Second Stage Optimization Results for Binary-Class Classification

Algorithm	Accuracy	F1 score	Time (Second)
XGBClassifier	0.99884	0.9988	3.362487793
RandomForestClassifier	0.99876	0.9988	9.394431114
BaggingClassifier	0.9986	0.9986	3.967015743

8.2.3 Third Stage Results

In the concluding phase, we fine-tuned the hyperparameters of the RF and XGBoost algorithms utilizing a genetic algorithm and the feature-reduced dataset. Table 8 provides a succinct comparison between the default hyperparameters and the values refined through genetic algorithm tuning. Additionally, Table 9 presents the noteworthy results derived from

Table 8 Best hyperparameters of XGBoost and RF with genetic algorithm after hybrid feature selection

Algorithms	Parameters	Default values	Best values
XGBoost	max_depth	6	3
	learning_rate	0.3	0.4564
	n_ estimators	100	333
	min_child_weight	1	3
	gamma	0	1.1631
	subsample	1	2.2589
	colsample_bytree	1	0.1548
	reg_alpha	1	0.0697
	reg_lambda	1	0.0019
RF	max_depth	None	20
	min_samples_leaf	1	1
	min_samples_split	2	2
	n_estimators	100	600

Table 9 Third Stage Optimization Results for Binary-Class Classification

Algorithm	Accuracy	F1 score	Time (Second)
XGBClassifier	0. 99,936	0.9995	16.84335375
RandomForestClassifier	0. 99,928	0.9994	20.43308616

this case study, indicating a substantial accuracy improvement for both algorithms—reaching 99.93% for RF and 99.93% for XGBoost compared to the preceding stage. Furthermore, the computational efficiency post-optimization has been halved compared to the initial stage, as demonstrated in Table 9. In Fig. 5, the confusion matrices of the optimized RF and XGBoost algorithms visually depict the positive impact of hyperparameter tuning, showcasing reduced false negative (FN) values and heightened accuracy. This outcome underscores the pronounced performance enhancements achieved through meticulous hyperparameter optimization.

9 Comparison Study

This section presents a comparative study to clearly illustrate the impact of feature selection and hyperparameter tuning on the accuracy of machine learning-based IDS detection. Our findings indicate that, for the multi-class classification case study, these processes did not yield notable improvements in the adopted IDS performance. However, a significant reduction in computational time was observed for all employed ML algorithms—a crucial metric for IDS systems operating in real-time scenarios. It has been observed that the proposed hybrid feature selection approach, coupled with the genetic algorithm for hyperparameter tuning, demonstrated nearly identical performance to using the original dataset with complete features but with a significantly reduced computational time. In conclusion, the proposed feature selection method with the genetic algorithm enhances the time efficiency of the IDS systems. Table 10 presents a comparative analysis of the F1 scores obtained during the experimental studies. Additionally, Table 11 provides a statistical representation

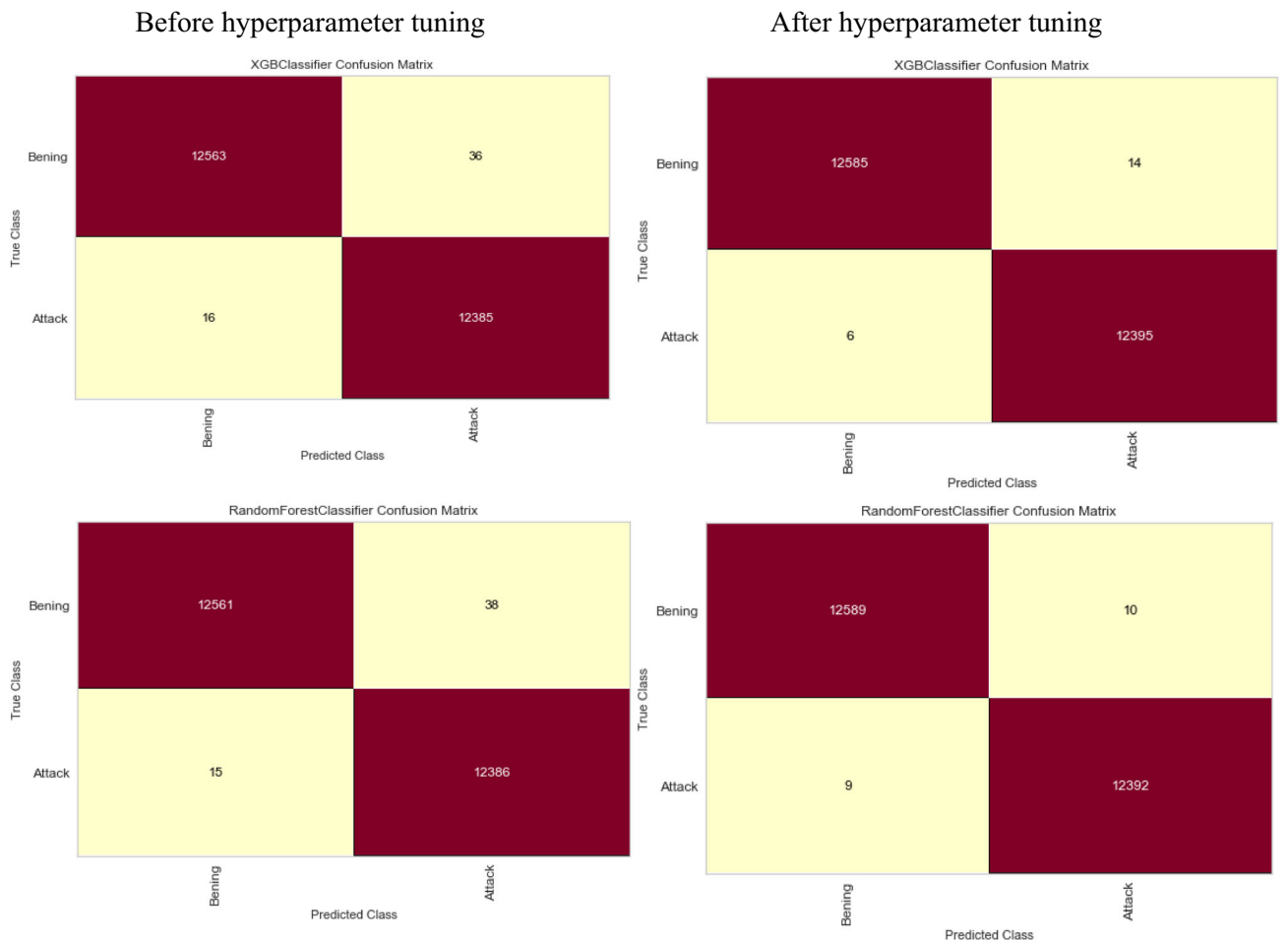

Fig. 5 Results of XGBoost and RF algorithms after applying the hybrid feature selection and GA algorithm for Binary Classification

Table 10 Results of all experiments

Used algorithm	Multi–Classification F1 score			Binary Classification F1 score		
	Default Value	After feature selection Default	After feature selection Hyper–parameter opt	Default Value	After feature selection Default	After feature selection Hyper–parameter opt
XGBoost	0.9998	0.9995	0.9998	0.9996	0.9988	0.9995
RF	0.9997	0.9995	0.9995	0.9995	0.9988	0.9994
Bagging	0.9997	0.9993	–	0.9994	0.9986	–
DT	0.9996	–	–	0.9992	–	–
ExtraTrees	0.9996	–	–	0.9991	–	–

Table 11 Time comparison

Model	Binary			Multi		
	Time in default case (Second)	Time after applying feature selection + GA (Second)	Time improvement (%)	Time in default case (Second)	Time after applying feature selection + GA (Second)	Time improvement (%)
XGBoost	32.94	16.84	48.87	70.15	42.02	40.15
RF	1470.25	20.43	98.61	891.36	6.57	99.26

highlighting a significant enhancement in the time efficiency of the IDS system subsequent to the application of the proposed optimization procedure.

10 Testing the Proposed Method Using Different Dataset

To validate the effectiveness of the proposed method, we applied it to an additional dataset, specifically the CSE-CIC 2018 dataset. Our focus was on training the optimized ML algorithm, namely XGBoost, with hyperparameter values selected using the genetic algorithm. We opted for a binary classification scenario over this dataset, initially labeling samples as either "attack" or "benign." Subsequently, we decoded, rescaled feature values, and pre-processed the data in the dataset. The proposed hybrid feature selection method was then employed to identify the most relevant features, resulting in the selection of three features to optimize the IDS system's detection accuracy. The final step involved using this selected feature dataset to train the XGBoost classifier with hyperparameter values chosen through the genetic algorithm. The achieved results demonstrated exceptional performance, with the proposed approach achieving 100% accuracy and F1 score, all within a highly competitive computational time. The results obtained by training the IDS system using this dataset is illustrated in Table 12. Moreover, the confusion matrix of the improved XGBoost is illustrated in Fig. 6.

Table 12 Third Stage Optimization Results for Binary-Class Classification

Algorithm	Accuracy	F1 score	Time (Second)
XGBClassifier	1.0	1.0	5.397483

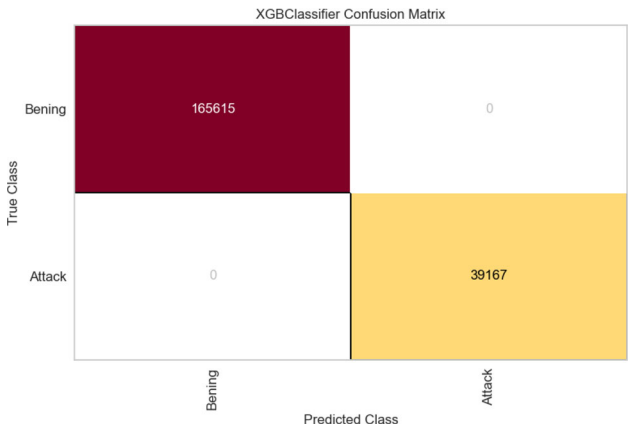

Fig. 6 The confusion matrix of the XGBoost trained using the CSE-CIC 2018 dataset

11 Conclusion

In conclusion, network attacks have detrimental effects on network performance and resource utilization, prompting the development of various methods for efficient attack detection. Among these methods, anomaly-based detection systems play a crucial role. This study focused on evaluating the performance of machine learning-based intrusion detection systems (IDSs) using the CICIDS2017 dataset. Our proposed hybridized IDS system comprises multiple stages. Initially, data pre-processing steps were implemented to cleanse the dataset and eliminate outlier points. Simultaneously, a hybridized data-balancing approach was introduced to address dataset imbalance. In the first stage, we assessed the performance of multiple machine learning algorithms with their default hyperparameters to evaluate their efficiency in attack detection. The subsequent stages involved proposing a hybridized feature selection method, integrating various feature selection techniques, and employing a genetic algorithm to fine-tune hyperparameters. These stages aimed to enhance IDS performance in both binary and multi-class classification tasks. The experimental results demonstrated that our hybrid feature selection method, coupled with hyperparameter optimization, notably improved the efficiency of XGBoost and RF algorithms, particularly in terms of computational time.

XGBoost consistently exhibited superior detection accuracy in both binary and multi-class classification applications. The hyperparameter tuning process, applied after feature selection, significantly reduced both false negatives and false positives, showcasing improvements of up to 61% and 62.5% for XGBoost, and 40% and 72.5% for random forest in binary classification.

The most significant achievement was the substantial reduction in overhead time, a critical metric for IDS systems. The proposed hybrid feature selection method, combined with genetic algorithm-based hyperparameter tuning, resulted in over 40% and 98% reduction in training time for XGBoost and RF-based IDS, respectively, in both binary and multi-class detection processes.

To validate the efficiency of our approach across diverse datasets, we tested it on the CSE-CIC 2018 dataset, achieving a 100% F1 score in detecting attacks. These findings have crucial implications for the development of effective IDS systems, enabling the identification of optimal hyperparameters and a reduction in feature dimensions for enhanced model efficiency and performance.

Looking ahead, future research could explore alternative hyperparameter optimization techniques and feature selection methods, along with assessing the performance of machine learning-based IDSs on different datasets. Additionally, experiments could be conducted to evaluate the impact of hyperparameter tuning and feature selection on the performance of deep learning models.

Acknowledgements The authors would like to thank Sharafaldin et al [59] for sharing their datasets.

Author Contributions Halit Bakır executed the experiment applications, visualizations, graphics, and article revisions. Özlem Ceviz authored the initial manuscript version and conducted the first set of experiments. Additionally, Halit Bakır provided supervision throughout the process.

Funding Open access funding provided by the Scientific and Technological Research Council of Türkiye (TÜBİTAK).

Availability of data and materials The dataset will be available on request.

Declarations

Conflict of interest Not applicable.

Ethical Approval Not applicable.

End Note This research was carried out during the 'Python for Artificial Intelligence (Python ile Yapay Zeka)' course at Sivas University of Science and Technology during the fall semester of 2022–2023.

References

1. Dave Smith, "IoT 2022: Connected devices growing 18% to 14.4 Billion globally," IOT For All, (2020)
2. Díaz López, D., et al.: Shielding IoT against cyber-attacks: an event-based approach using SIEM. Wirel. Commun. Mob. Comput. (2018). https://doi.org/10.1155/2018/3029638
3. Sicari, S.; Rizzardi, A.; Miorandi, D.; Coen-Porisini, A.: REATO: REActing TO denial of service attacks in the internet of things. Comput. Netw. **137**, 37–48 (2018). https://doi.org/10.1016/j.comnet.2018.03.020
4. Dave Irvine, "Report shows 118 percent increase in ransomware attacks In 2019," Sep. (2019)
5. Pawar, A.B.; Ghumbre, S.: A survey on IoT applications, security challenges and counter measures. Int. Conf. Comput. Anal. Secur. Trends CAST **2016**, 294–299 (2017). https://doi.org/10.1109/CAST.2016.7914983
6. Modi, C., Patel, D., Borisaniya, B., Patel, H., Patel, A., Rajarajan, M.: A survey of intrusion detection techniques in cloud. J. Netw. Comput. Appl. **36**(1), 42–57 (2013). https://doi.org/10.1016/j.jnca.2012.05.003

7. Mishra, P.; Varadharajan, V.; Tupakula, U.; Pilli, E.S.: A detailed investigation and analysis of using machine learning techniques for intrusion detection. IEEE Commun. Sur. Tutor. **21**(1), 686–728 (2019). https://doi.org/10.1109/COMST.2018.2847722
8. Masduki, B. W.; Ramli, K.; Saputra, F. A.; Sugiarto, D.: Study on implementation of machine learning methods combination for improving attacks detection accuracy on intrusion detection system (IDS), in 2015 International Conference on Quality in Research (QiR), IEEE, pp. 56–64 (2015)
9. Al-Garadi, M.A.; Mohamed, A.; Al-Ali, A.K.; Du, X.; Ali, I.; Guizani, M.: A Survey of machine and deep learning methods for internet of things (IoT) security. IEEE Commun. Surv. Tutor. **22**(3), 1646–1685 (2020). https://doi.org/10.1109/COMST.2020.2988293
10. Feurer, M.; Hutter, F.: "Hyperparameter optimization," Automated machine learning: Methods, systems, challenges, pp. 3–33, (2019)
11. Kunang, Y.N.; Nurmaini, S.; Stiawan, D.; Suprapto, B.Y.: Attack classification of an intrusion detection system using deep learning and hyperparameter optimization. J. Inf. Secur. Appl. **58**, 102804 (2021)
12. Maseer, Z.K.; Yusof, R.; Bahaman, N.; Mostafa, S.A.; Foozy, C.F.M.: Benchmarking of machine learning for anomaly based intrusion detection systems in the CICIDS2017 dataset. IEEE Access **9**, 22351–22370 (2021). https://doi.org/10.1109/ACCESS.2021.3056614
13. Doğan, E.; H. Bakir, H.: "Hiperparemetreleri Ayarlanmış Makine Öğrenmesi Yöntemleri Kullanılarak Ağdaki Saldırıların Tespiti," in International Conference on Pioneer and Innovative Studies, pp. 274–286 (2023)
14. Yousefnezhad, M.; Hamidzadeh, J.; Aliannejadi, M.: Ensemble classification for intrusion detection via feature extraction based on deep learning. Soft comput **25**(20), 12667–12683 (2021). https://doi.org/10.1007/s00500-021-06067-8
15. Sharma, D.K.; Mishra, J.; Singh, A.; Govil, R.; Srivastava, G.; Lin, J.C.W.: Explainable artificial intelligence for cybersecurity. Comput. Electr. Eng. (2022). https://doi.org/10.1016/j.compeleceng.2022.108356
16. Priyanka, V.; Gireesh Kumar, T.: Performance assessment of IDS based on CICIDS-2017 dataset. Lect. Notes Net. Syst. **191**, 611–621 (2020). https://doi.org/10.1007/978-981-16-0739-4_58
17. Bakour, K.; Daş, G.S.; Ünver, H.M.: "An intrusion detection system based on a hybrid Tabu-genetic algorithm. Int. Conf. Comput. Sci. Eng. (UBMK) (2017). https://doi.org/10.1109/UBMK.2017.8093378
18. Hossain, M. D.; Ochiai, H.; Fall, D.; Kadobayashi, Y.: "LSTM-based network attack detection: performance comparison by hyper-parameter values tuning," in 2020 7th IEEE International Conference on Cyber Security and Cloud Computing (CSCloud)/2020 6th IEEE International Conference on Edge Computing and Scalable Cloud (EdgeCom), IEEE pp. 62–69 (2020)
19. Kim, M.: Supervised learning-based DDoS attacks detection: Tuning hyperparameters. ETRI J. **41**(5), 560–573 (2019). https://doi.org/10.4218/etrij.2019-0156
20. Batchu, R.K.; Seetha, H.: A generalized machine learning model for DDoS attacks detection using hybrid feature selection and hyperparameter tuning. Comput. Net. **200**, 108498 (2021). https://doi.org/10.1016/j.comnet.2021.108498
21. Choraś, M.; Pawlicki, M.: Intrusion detection approach based on optimised artificial neural network. Neurocomputing **452**, 705–715 (2021). https://doi.org/10.1016/j.neucom.2020.07.138
22. Sanchez, O. R.; Repetto, M.; Carrega, A.; Bolla, R.: "Evaluating ML-based DDoS detection with grid search hyperparameter optimization," in 2021 IEEE 7th International Conference on Network Softwarization (NetSoft), IEEE, pp. 402–408 (2021)
23. Kunang, Y. N.; Nurmaini, S.; Stiawan, D.; Suprapto, B. Y.: "Improving Classification attacks in IOT intrusion detection system using bayesian hyperparameter optimization," 2020 3rd international seminar on research of information technology and intelligent systems, ISRITI pp. 146–151 (2020) https://doi.org/10.1109/ISRITI51436.2020.9315360
24. Okey, O.D.; Melgarejo, D.C.; Saadi, M.; Rosa, R.L.; Kleinschmidt, J.H.; Rodríguez, D.Z.: Transfer learning approach to IDS on cloud IoT devices using optimized CNN. IEEE Access **11**, 1023–1038 (2023)
25. Sharma, B.; Sharma, L.; Lal, C.; Roy, S.: Anomaly based network intrusion detection for IoT attacks using deep learning technique. Comput. Electr. Eng. **107**, 108626 (2023). https://doi.org/10.1016/j.compeleceng.2023.108626
26. Bakhshad, S.; Ponnusamy, V.; Annur, R.; Waqasyz, M.; Alasmary, H.; Tux, S.: "Deep Reinforcement learning based intrusion detection system with feature selections method and optimal hyperparameter in IoT environment," International Conference on Computer, Information and Telecommunication Systems (CITS), 2022, pp. 1–7. doi: https://doi.org/10.1109/CITS55221.2022.9832976.
27. Saurabh, K., et al.: "Lbdmids: LSTM based deep learning model for intrusion detection systems for IOT networks", in. IEEE World AI IoT Congress (AIIoT) **2022**, 753–759 (2022)
28. Mohy-eddine, M.; Guezzaz, A.; Benkirane, S.; Azrour, M.: An efficient network intrusion detection model for IoT security using K-NN classifier and feature selection. Multimed Tools Appl **82**(15), 1–19 (2023)
29. Mishra, D.; Naik, B.; Nayak, J.; Souri, A.; Dash, P.B.; Vimal, S.: Light gradient boosting machine with optimized hyperparameters for identification of malicious access in IoT network. Digit. Commun. Net. **9**(1), 125–137 (2023)
30. Manzano, R.; Goel, N.; Zaman, M.; Joshi, R.; Naik, K.: "Design of a machine learning based intrusion detection framework and methodology for iot networks," in 2022 IEEE 12th Annual computing and communication workshop and conference (CCWC), pp. 191–198 (2022)
31. Hossain, M. D.; Ochiai, H.; Fall, D.; Kadobayashi, Y.: "LSTM-based network attack detection: performance comparison by hyper-parameter values tuning," Proceedings–2020 7th IEEE International conference on cyber security and cloud computing and 2020 6th IEEE International conference on edge computing and scalable cloud, CSCloud-EdgeCom 62–69 (2020) https://doi.org/10.1109/CSCloud-EdgeCom49738.2020.00020.
32. Sanchez, O. R.; Repello, M.; Carrega, A.; Bolla, R.: "Evaluating ML-based DDoS detection with grid search hyperparameter optimization," Proceedings of the 2021 IEEE conference on network softwarization: accelerating network softwarization in the cognitive age, NetSoft, no. Ml, pp. 402–408, (2021) https://doi.org/10.1109/NetSoft51509.2021.9492633.
33. Bakır, H., Bakır, R.: DroidEncoder: malware detection using auto-encoder based feature extractor and machine learning algorithms. Comput. Electr. Eng. **110**, 108804 (2023)
34. Bakır, H., Elmabruk, K.: Deep learning-based approach for detection of turbulence-induced distortions in free-space optical communication links. Phys. Scr. **98**(6), 065521 (2023)
35. Demircioğlu, U.; Bakır, H.: Deep learning-based prediction of delamination growth in composite structures: bayesian optimization and hyperparameter refinement. Phys. Scr. **98**(10), 106004 (2023)
36. Bakir, H.; Yilmaz, Ş: Using Transfer learning technique as a feature extraction phase for diagnosis of cataract disease in the eye. Int. J. Sivas Univ. Sci. Technol. **1**(1), 17–33 (2022)
37. Yilmaz, E. K.; Bakir, H.: "Hyperparameter Tunning and feature selection methods for malware detection," Politeknik Dergisi, p. 1, (2023)
38. Bakir, H.; Oktay, S.; Tabaru, E.: Detection of pneumonia from x-ray images using deep learning techniques. J. Sci. Rep.-A **052**, 419–440 (2023)

39. Bakır, H.; Çayır, A. N.; Navruz, T. S.: "A comprehensive experimental study for analyzing the effects of data augmentation techniques on voice classification," Multimed Tools Appl, pp. 1–28, (2023)
40. Bakir, H.; Bakir, R.: Evaluating the robustness of yolo object detection algorithm in terms of detecting objects in noisy environment. J. Sci. Rep.-A **054**, 1–25 (2023)
41. H. Bakır, H."Evaluating the impact of tuned pre-trained architectures' feature maps on deep learning model performance for tomato disease detection," Multimed Tools Appl, pp. 1–22, 2023.
42. Demircioğlu, U.; Bakır, H.: Deep learning-based prediction of delamination growth in composite structures: bayesian optimization and hyperparameter refinement. Phys. Scr. **98**(10), 106004 (2023)
43. Bakir, H.; Eker, S. B.: "A comprehensive experimental study for evaluating the performance of well-known cnn pre-trained models in noisy environments," Politeknik Dergisi, p. 1 (2023)
44. Ghanem, R.; Erbay, H.: Context-dependent model for spam detection on social networks. SN Appl Sci **2**, 1–8 (2020)
45. Ghanem, R.; Erbay, H.: Spam detection on social networks using deep contextualized word representation. Multimed Tools Appl **82**(3), 3697–3712 (2023)
46. Quinlan, J.R.: Induction of decision trees. Mach. Learn. **1**(1), 81–106 (1986). https://doi.org/10.1007/bf00116251
47. Hasan, M.; Islam, M.M.; Zarif, M.I.I.; Hashem, M.M.A.: Attack and anomaly detection in IoT sensors in IoT sites using machine learning approaches. Int. Things (Netherlands) **7**, 100059 (2019). https://doi.org/10.1016/j.iot.2019.100059
48. Breiman, L.: Random forests. Mach. Learn. **45**, 5–32 (2001)
49. Ariyaluran Habeeb, R.A.; Nasaruddin, F.; Gani, A.; Targio Hashem, I.A.; Ahmed, E.; Imran, M.: Real-time big data processing for anomaly detection: a Survey. Int. J. Inf. Manage. **45**, 289–307 (2019). https://doi.org/10.1016/j.ijinfomgt.2018.08.006
50. Dhaliwal, S.S.; Al Nahid, A.; Abbas, R.: Effective intrusion detection system using XGBoost. Information (Switzerland) (2018). https://doi.org/10.3390/info9070149
51. Bhati, B.S.; Chugh, G.; Al-Turjman, F.; Bhati, N.S.: An improved ensemble based intrusion detection technique using XGBoost. Trans. Emerg. Telecommun. Technol. **32**(6), 1–15 (2021). https://doi.org/10.1002/ett.4076
52. Tsuruoka,Y.; Tsujii, J.; Ananiadou, S.: "Stochastic gradient descent training for L1-regularized log-linear models with cumulative penalty," ACL-IJCNLP 2009 - Joint Conf. of the 47th annual meeting of the association for computational linguistics and 4th Int. Joint Conf. on natural language processing of the AFNLP, Proceedings of the Conf., pp. 477–485, (2009) https://doi.org/10.3115/1687878.1687946.
53. Sulaiman, M. A.; Labadin, J.: "Feature selection based on mutual information," in 2015 9th International conference on IT in Asia (CITA), IEEE, (2015) pp. 1–6
54. Li, J., et al.: Feature selection: a data perspective. ACM Comput. Surv. (2017). https://doi.org/10.1145/3136625
55. Holland, J.H.: Genetic algorithms. Sci. Am. **267**(1), 66–73 (1992)
56. Singh, T.; Verma, S.; Kulshrestha, V.; Katiyar, S.: Intrusion detection system using genetic algorithm for cloud. ACM Int. Conf. Proc. Ser. **04**, 564–568 (2016). https://doi.org/10.1145/2905055.2905175
57. Sazzadul Hoque, M.: An implementation of intrusion detection system using genetic algorithm. Int. J. Net. Secur. Appl. **4**(2), 109–120 (2012). https://doi.org/10.5121/ijnsa.2012.4208
58. Alibrahim, H.; Ludwig, S. A.: "Hyperparameter optimization: comparing genetic algorithm against grid search and bayesian optimization," 2021 IEEE Congress on evolutionary computation, CEC 2021–Proceedings, pp. 1551–1559, (2021) https://doi.org/10.1109/CEC45853.2021.9504761.
59. Sharafaldin, I.; Lashkari, A. H.; Ghorbani, A. A.: "Toward generating a new intrusion detection dataset and intrusion traffic characterization," ICISSP 2018–Proceedings of the 4th International Conference on Information Systems Security and Privacy, Cic, pp. 108–116, (2018) https://doi.org/10.5220/0006639801080116.