IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 27 May 2023, accepted 30 June 2023, date of publication 4 July 2023, date of current version 17 July 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3292267

RESEARCH ARTICLE

Enhancing Intrusion Detection in IoT Communications Through ML Model Generalization With a New Dataset (IDSAI)

GUTIERREZ-PORTELA FERNANDO[1], **ARTEAGA-ARTEAGA HAROLD BRAYAN**[2], **ALMENARES MENDOZA FLORINA**[3], **(Member, IEEE), CALDERÓN-BENAVIDES LILIANA**[4], **ACOSTA-MESA HÉCTOR-GABRIEL**[5], **AND TABARES-SOTO REINEL**[2,6,7]

[1]Grupo de Investigación Aqua, Universidad Cooperativa de Colombia, Ibagué, Tolima 730006, Colombia
[2]Departamento de Electrónica y Automatización, Universidad Autónoma de Manizales, Manizales, Caldas 170001, Colombia
[3]Departamento de Ingeniería Telemática, Universidad Carlos III de Madrid (UC3M), 28911 Madrid, España
[4]Unidad Académica Tecnologías de la Información, Universidad Autónoma de Bucaramanga (UNAB), Bucaramanga, Santander 680003, Colombia
[5]Instituto de Investigaciones en Inteligencia Artificial, Universidad Veracruzana, Xalapa, Veracruz 91000, Mexico
[6]Departamento de Sistemas e Informática, Universidad de Caldas, Manizales, Caldas 170004, Colombia
[7]Facultad de Ingeniería y Ciencias, Universidad Adolfo Ibáñez, Santiago 7941169, Chile

Corresponding author: Tabares-Soto Reinel (rtabares@autonoma.edu.co)

This work was supported in part by Universidad Cooperativa de Colombia under Project INV2658-I; in part by the Ministry of Economy and Competitiveness, Spain, through the Research Project COMPROMISE under Grant PID2020-113795RBC32/AEI/10.13039/501100011033; and in part by Universidad Autónoma de Manizales contributed to this work through the Application of Convolutional Neural Networks for Steganalysis Project under Grant 645-2019 TD.

ABSTRACT One of the fields where Artificial Intelligence (AI) must continue to innovate is computer security. The integration of Wireless Sensor Networks (WSN) with the Internet of Things (IoT) creates ecosystems of attractive surfaces for security intrusions, being vulnerable to multiple and simultaneous attacks. This research evaluates the performance of supervised ML techniques for detecting intrusions based on network traffic captures. This work presents a new balanced dataset (IDSAI) with intrusions generated in attack environments in a real scenario. This new dataset has been provided in order to contrast model generalization from different datasets. The results show that for the detection of intruders, the best supervised algorithms are XGBoost, Gradient Boosting, Decision Tree, Random Forest, and Extra Trees, which can generate predictions when trained and predicted with ten specific intrusions (such as ARP spoofing, ICMP echo request Flood, TCP Null, and others), both of binary form (intrusion and non-intrusion) with up to 94% of accuracy, as multiclass form (ten different intrusions and non-intrusion) with up to 92% of accuracy. In contrast, up to 90% of accuracy is achieved for prediction on the Bot-IoT dataset using models trained with the IDSAI dataset.

INDEX TERMS Deep learning, internet of things, intrusion detection system, machine learning, wireless sensor network.

I. INTRODUCTION

The deployment in the interconnection of the Internet of Things (IoT) and Wireless Sensor Networks (WSN) has taken relevance thanks to their contribution to the development of smart cities in domains such as transportation, mobility, economy, industry, health, among others [1]. Most of these domains require processing capabilities closer to where the data originates. According to IoT Analytics, there are expected to be more than 30 billion IoT connections by 2025, corresponding to four IoT devices per person [2]. Also, with the exponential growth of IoT technology solutions connected to the cloud, new security and privacy threats related to data and services make them an attractive surface for intrusions. In the same way, network security can be threatened by limited resources such as storage capacity, processing speed,

The associate editor coordinating the review of this manuscript and approving it for publication was Shaohua Wan.

memory limitations, the power of end devices, and the use of wireless communications by hosts (which are vulnerable due to their ease of access) [3], [4].

Wireless networks are more vulnerable to attacks due to their transmission medium, which poses a challenge to existing security mechanisms in their attempt to mitigate emerging threats. For this reason, a number of different solutions have been proposed in the academic literature [5]. For example, efficient autonomous defense systems have been proposed that use machine learning techniques in devices at the perimeter of the network [6]. Similarly, creating an intelligent cybersecurity support architecture has been explored [7], as well as using Machine Learning-based resource management techniques in fog computing platforms [8]. Other approaches include intrusion detection and prevention systems applied to new trends and applications in IoTs and related areas like WSNs, Mobile Ad Hoc Network (MANET), and Connection Point Services (CPS) [4]. Intrusion detection and prevention systems are considered a second line of defense. However, as new attack techniques emerge, it is necessary to develop systems with optimal performance and low resource consumption [9], [10].

Researchers have used anomaly-based network intrusion detection models with Deep Learning (DL) usage in airports [11], intrusion detection models for cyber security in Agriculture 4.0 [12] and to prevent DoS attack in WSN an edge intelligence framework [13], they have identified malicious traffic with anomaly detection techniques and DL detection systems with Auto-encoders [14].

Some difficulties are carried due to the high level of complexity and high consumption of computational resources by detection systems deployed in networks [9], and also due to the low reliability in the quality and accuracy of collected data, and loss of services and information [15].

Unauthorized incursions into the system are called intrusions or attacks. A user can intrude internally or externally. In an internal attack, the user with privileged access obtains restricted information and gains control of the system or network. The external intruder seeks permission to arbitrarily access the system or network to enter and steal vital information from a company and gain control of the system or network [16]. The main functions to be performed by an Intrusion Detection System (IDS) include: i) identifying an intruder, ii) notifying the location of an attacker, iii) logging abnormal movements, iv) minimizing or interrupting malicious actions, e) alerting the administrator of the security intrusions, and v) detecting the type of intrusion [10].

Accordingly, the issues mentioned above, the design and implementation of an anomaly-based IDS employing ML continue to be addressed and evolved. The pipelines must include a Network Configuration that must be established in an environment to send particular attacks in a controlled way. The network traffic data is captured to create a dataset. The dataset is divided into training and testing, with which ML models are trained, and validations are made. At this point, it is already possible to detect and report intrusions, with which, according to the defined control mechanisms, decisions are made, and alarms are generated.

Some advances in the field are through automatic classification techniques that are increasingly accurate in identifying abnormal patterns or anomalies in IDS modeling to reduce the false alarm rate [17]. The development of datasets (Bot-IoT) for network forensics [18], building ML models to identify IoT network attacks [19], the IDS and the comparison of ML classifiers [20], intrusion detection models with supervised and unsupervised algorithms [21], ML models in anomaly-based IDS using the CICIDS2017 and the NSL-KDD datasets [22], [23].

This study contributes significant advancements to the field of cybersecurity in IoT networks from various perspectives. Firstly, it introduces a new dataset called *Intrusion Detection System Artificial Intelligence* (IDSAI), obtained in a real and balanced attack environment. Secondly, it compares the classification capabilities of an IDS based on machine learning for detecting attacks in an IoT system, evaluating the performance of eight machine learning algorithms, including Extreme Gradient Boosting or XGBoost (XGB), Gradient Boosting (GB), Decision Tree (DT), Random Forest (RF), and Extra Trees (ET). These algorithms are experimented with in three different scenarios to select the most effective ones. Also, essential feature selection techniques are employed to enhance the classification process. Thirdly, explanatory artificial intelligence is utilized to provide insights into the Machine Learning (ML) classification models and identify the most relevant features for each intrusion class. Finally, the proposed system is evaluated through cross-validation of datasets.

The main contributions of this research can be summarized as follows:

- Generation of a novel and balanced dataset (IDSAI) obtained from a real-based attack setting. The IDSAI dataset includes ten different types of intrusions and non-intrusion data, providing a valuable resource for intrusion detection research in IoT networks, enabling an accurate evaluation of intrusion detection algorithms. On the other hand, the IDSAI data set encourages research and comparison of results between different studies, thus contributing to the security and protection of IoT networks in the real world.
- Comparison of the classification performance of our proposed Intrusion Detection System across binary and multiclass scenarios using eight machine learning algorithms. This analysis enables the evaluation and comparison of machine learning models' effectiveness in detecting and classifying attacks on IoT systems. By identifying the most efficient algorithms, the response capacity and protection of IoT systems against threats can be enhanced, achieving accurate detection of attacks.
- Study of essential features for binary classification and by attacks (multiclass classification) using machine

TABLE 1. Comparison of novel works, in terms of data, ML algorithms used, metrics, types of attacks and their main contributions.

Author	Dataset	Model	Accuracy [%]	Recall [%]	F1-score [%]	Precision [%]	Attack Type	Principal contributions
[17]	KDD’99	KNN	96.55	93.67	-	-	DoS, U2R, R2L, and Probe	Improvements in detection speed and accuracy. Effective clustering rules for network-based mining methods.
		K-Means	92.30	91.58	-	-		
[20]	UNSW-NB15	LR	70.00	-	54.00	90.00	Binary: Normal and Attack. Multiclass: Analysis, Backdoor, DoS, Exploits, Fuzzers, Generic, Reconnaissance, Shellcode, and Worms.	Random Forest is the best for IDS with an accuracy of 87%, precision of 98%, and F1-score of 84%.
		MultinomialNB	75.00	-	76.00	68.00		
		GNB	71.00	-	62.00	75.00		
		KNN	78.00	-	75.000	87.00		
		DT	86.00	-	83.00	93.00		
		RF	87.00	-	84.00	98.00		
		MLP	75.00	-	73.00	71.00		
		GB	86.00	-	82.00	98.00		
[21]	NSL-KDD	LR	72.25	91.94	74.05	61.99	Binary: Normal and Attack. Multiclass:. BENIGN, DoS slowloris, DoS Slowhttptest, DoS Hulk, DoS GoldenEye, Heartbleed, PortScan, DDoS, FTP-Patator, SSH-Patator, DoS Slow HTTP Test, Bot, Web Attack-Brute Force, Web Attack- XSS, Infiltration, Web Attack-Sql Injection (Fifteen different types of attacks are analyzed)	The study presents several balanced and unbalanced datasets as benchmarks for NSLKDD and CICIDS2017 evaluations. The implementations reveal that Random Forest and K-Means outperform the other approaches for supervised learning to address network protection threats.
		DT	75.45	96.75	77.25	64.29		
		RF	76.50	97.22	78.09	65.25		
		GNB	74.34	86.90	74.47	65.16		
		KNN	76.41	96.26	77.85	65.36		
	CICIDS2017	LR	82.30	65.42	59.21	54.08		
		DT	89.16	94.73	77.44	65.48		
		RF	93.77	84.15	84.15	84.15		
		GNB	69.67	47.19	37.93	31.70		
		KNN	90.69	98.41	80.59	68.23		
[22]	CICIDS2017	ANN	99.00	99.00	99.00	99.00	Binary: Normal and Abnormal. Multiclass: BENIGN (C1) Brute Force (C2) XSS (C3) SQL Injection (C4)	Uses real data to ensure practical evaluation of the performance of AIDS based on ML algorithms. A single ML algorithm that can detect all types of web attacks. The best models are KNN, DT, and NB.
		DT	99.00	99.00	99.00	99.00		
		k-NN	99.00	99.00	99.00	99.00		
		NB	99.00	99.00	99.00	99.00		
		RF	99.00	99.00	99.00	99.00		
		SVM	75.00	75.00	76.00	99.00		
		CNN	99.00	99.00	99.00	99.00		
		k-MEANS	25.00	25.00	40.00	97.00		
		EM	60.00	60.00	74.00	86.00		
		SOM	59.00	60.00	74.00	86.00		
[23]	NSL-KDD	SGD	99.31	99.24	99.12	98.88	DoS	Feature selection. Multi ML model evaluation. Demostration of that RF is good for training attacks, however RF has not good performance in new attacks. Demostration of that DT presents favorable results in identification of news attacks.
			99.22	96.95	97.17	97.43	Probe	
			96.54	95.49	94.69	94.06	R2L	
			99.56	81.90	81.85	90.71	U2R	
		RC	97.81	96.78	97.42	98.08	DoS	
			97.99	98.14	96.94	95.85	Probe	
			96.11	95.88	94.65	93.56	R2L	
			99.67	82.93	86.28	92.90	U2R	
		DT	99.63	99.66	99.58	99.50	DoS	
			99.57	99.26	99.32	99.39	Probe	
			97.92	96.95	97.05	97.15	R2L	
			99.65	90.95	88.21	86.29	U2R	
		RF	99.83	99.69	99.80	99.92	DoS	
			99.67	99.24	99.46	99.58	Probe	
			98.04	96.86	97.14	97.33	R2L	
			99.76	82.10	90.88	96.38	U2R	
		ET	99.79	99.74	99.75	99.77	DoS	
			99.65	99.31	99.45	99.59	Probe	
			97.93	96.93	97.06	97.20	R2L	
			99.83	93.17	94.02	95.75	U2R	
[24]	Bot-IoT	DT	99.99	100.0	-	100.0	DDoS TCP, DDoS UDP, DoS TCP, DoS UDP, Reconnaissance OS Fingerprint, Reconnaissance Service Scan	ML algorithm comparison for intrusion detection.
		GNB	99.79	98.00	-	99.00		
		RF	99.99	100.0	-	100.0		
[25]	Bot-IoT	DT	98.40	-	-	-	DDoS TCP, DDoS UDP, DoS TCP, DoS UDP, Reconnaissance OS Fingerprint, Reconnaissance Service Scan	ML algorithms can predict better and faster than some DL approaches.
		RF	97.10	-	-	-		
		DL	69.30	-	-	-		
[26]	CUPID	RF	97.66	80.42	88.93	99.45	Webcrawling, Recorded live user interaction, ARP, nmap, Dig, DNSMap, DNSTracer, nslookup, SQLi, Directory Traversal, Password brute forcing, Delivery of reverse Meterpreter shell, STP, DHCP attacks, BoNeSi	The study presents a new University of Colorado Pentesting (CUPID) intrusion dataset.
		KNN	99.37	96.81	97.30	97.78		
		MLP	99.40	96.50	97.41	98.33		
[27]	AWID	SVM	97.44	99.98	81.38	69.90	Normal, Injection, Impersonation, Fooding	Improving intrusion detection in smart city applications with two feature selection techniques and comprehensive metric-based comparisons.
		RF	98.60	84.18	88.62	92.38		
		DT	98.23	85.07	89.39	94.47		
[28]	RPL-NIDDS17	Voting	98.00	94.00	95.00	95.00	Clone ID, Hello flooding, Local repair, Selective forwarding, Sinkhole, Blackhole and Sybil.	Method for intrusion detection using feature extraction and selection techniques, as well as a weighted majority voting classifier.
		AB	94.00	91.00	91.00	91.00		
		B	92.00	88.00	88.00	87.00		

learning techniques. By studying these features, we gain insights into the key factors that contribute to accurate intrusion detection, enhancing our understanding of the underlying patterns and characteristics of different types of attacks.

- Evaluation of the generalization power of the models through a cross-validation strategy that shows the effectiveness of attack detection models in IoT networks. We assess the performance of the trained models by making predictions on the Bot-IoT dataset, achieving an unbiased evaluation greater than 90% in scenario 3 when using models trained with the IDSAI dataset. This supports the robustness and effectiveness of the proposal in the detection of attacks in real-world IoT networks with a significant impact on improving the security and protection of IoT networks against threats and attacks. This validation approach demonstrates the robustness and effectiveness of our proposed approach in detecting attacks in real-world IoT networks.

This study addresses the problem of intrusion detection in IoT network traffic by introducing the IDSAI dataset, conducting comparative analysis of IDS classification performance, exploring essential features, and evaluating model generalization. Our findings contribute to the field of cybersecurity in IoT networks and have practical implications for enhancing network security.

The remaining sections of this paper have the following order: Section II describes related work, which uses mainly public datasets. Section III explains the proposed dataset, methodology, models, and metrics used in this work. Section IV presents the main results obtained for classifying intrusions and discussing them. Finally, Section V shows the conclusions and the future work.

II. RELATED WORKS

The following are studies in applying anomaly-based detection techniques using ML to identify security intrusions in IoT networks. **Table 1** shows a comparison of the related works.

Reference [17] implemented techniques such as K-Nearest Neighbors (KNN) using the Decision Tree Method (DTM) and K-Means to reduce the false alarm rate in the IDS, with the KDD’99 dataset. KNN achieve an accuracy of 96.55% and a Recall of 93.67%. The k-Means model obtains an accuracy of 92.30% and a Recall of 91.58%.

Reference [20] compared Logistic Regression (LR), Multinomial Naive Bayes (MultinomialNB), Gaussian Naive Bayes (GNB), KNN, DT, RF, MLP, and GB classifiers. The metrics to validate the binary and multiclass scenarios are accuracy, precision, and F1-score. The dataset used in the experiment was UNSW-NB15. The results showed that the Random Forest classifier outperforms the other models in terms of accuracy at 87%, precision at 98%, and F1-score at 84%.

Reference [21] compared supervised learning models with NSL-KDD and CICIDS2017 datasets. In this study, in the NSL-KDD dataset, the RF and KNN algorithms generated the best performances with accuracy, recall, F1-score, and precision up to approximately 76%, 96%, 77%, and 65%, respectively. With the CICIDS2017 dataset, RF achieves an accuracy of up to 93%, recall, F1-score, and precision of up to 84%.

Reference [22] carefully reviewed research on IDS with AI and employed supervised ML algorithms, which included Artificial Neural Network (ANN), DT, KNN, Naive Bayes, RF, SVM, Convolutional Neural Network (CNN), K-Means, Expectation-Maximization (EM), and Self Organizing Map (SOM) algorithms. For the experiment, they used the highly imbalanced multiclass CICIDS2017 dataset. As a result, they obtained that KNN, DT, and Naive Bayes models are the best for intrusion detection for the CICIDS2017 dataset (99% of accuracy). It is possible to detect all web attacks using a single algorithm.

Reference [23] presents the application of ML models such as SGD, Ridge Classifier (RC), DT, RF, and ET. The goal is the prediction of DoS, Probe, R2L, and U2R attacks using the NSL-KDD dataset. A feature selection process is carried out, and the DT for identifying news attacks is determined as a good alternative. The experiments with the ET and RF models achieve an accuracy of 99.83% using multiclass classification to detect U2R and DoS attacks, respectively.

Reference [24] proposed ML models to detect intrusion anomalies in IoT network traffic using the BoT-IoT dataset. They selected algorithms such as DT, GNB, and RF. The GNB algorithm is effective for the detection of intrusions.

Reference [25] proposed an IDS based on a big data platform that can differentiate between the types of network traffic flow generated by IoT devices. This work compares ML algorithms on the Apache Spark platform and found that ML algorithms outperform DL algorithms with higher accuracy and less training time for the model. The experimentation is carried out using the BoT-IoT real-world network traffic dataset.

Reference [26] designed a framework to gather data from the publicly available CUPID dataset, which had been annotated with human pentesting activity on the network. This framework facilitated the distinction between automatically generated attacks and those initiated by humans, at the feature level. The types of attacks generated included Webcrawling, Recorded live user interaction, ARP, nmap, Dig, DNSMap, DNSTracer, nslookup, SQLi, Directory Traversal, Password brute forcing, Delivery of reverse Meterpreter shell, STP, and DHCP attacks. For their analysis, the researchers employed supervised algorithms such as RF, KNN, MLP, and others.

Reference [27] proposed a novel method to detect injection attacks in IoT applications by leveraging feature selection and machine learning techniques. The researchers used the public AWID dataset and applied two feature selection techniques: constant deletion and recursive deletion. They used three machine learning algorithms for their analysis: SVM, Random Forest, and Decision Tree. This work suggests that appropriate feature selection can significantly enhance the accuracy of a model's attack detection capabilities in IoT applications.

Reference [28] proposed a comprehensive method encompassing preprocessing steps, SMOTE oversampling, feature extraction, feature selection, and a voting classifier. The chosen features were then classified using AB, B and Voating.

Studies above have applied ML techniques for intrusion detection systems on datasets such as Bot-IoT, KDD'99, CICIDS2017, UNSW-NB15, NSL-KDD, CUPID, Aegean Wifi Intrusion Dataset (AWID), and RPL-NIDDS17.

To provide a comprehensive overview of ML applications in stroke management, our research aligns with relevant studies in the field. For example, the article [29] presents an explainable AI model that utilizes ML techniques to predict acute strokes using EEG signals. Similarly, [30] introduces a cyber-physical system that utilizes ECG data to classify stroke patients with altered cardiac activity, facilitating real-time data processing and utilization for stroke identification and post-stroke treatment management. Additionally, [31] focuses on the utilization of a portable EEG device for real-time health monitoring and providing early prognostic information for stroke management. These studies collectively illustrate the wide-ranging applications of ML in stroke prediction, cardiac monitoring, and real-time health monitoring, complementing our research in IoT network cybersecurity.

These works implement IDS with the use of highly unbalanced datasets. The imbalance is caused by the nature of the problem since there are attacks less common than others, and, in general, there are more samples without attacks. Since ML models interpret the complexity and heterogeneity of the data, this search for patterns will be biased with unbalanced databases. They were making it necessary to release a balanced dataset with attacks, not synthetic ones obtained through repetitive or approximation balance techniques.

III. MATERIALS AND METHODS

In this section, the materials and methods employed in the study are described. The IDS architecture, which includes the physical system and its general structure, is outlined. The data used in the analysis, including the features and the different intrusions or classes, is presented. Additionally, the Bot-IoT dataset is utilized for testing purposes. The models employed and their training process, including hyperparameter tuning, are explained. The performance of the IDS is assessed, and the importance of features is determined. Finally, the resources utilized in the study are disclosed.

A. IDS ARCHITECTURE

1) PHYSICAL SYSTEM

The articulated system consists of devices, a network, and a cloud. The hardware elements used are: sensors, Arduino Nano V3.0 A, XBee-Pro S2C 2.4GHz Serie2 63mW (18dBm) communication devices, which achieve a data transmission rate of 250Kbps and comply with the 802.15.4 ZigBee standard, and finally, the Raspberry Pi 3 Model B+ integration platform, which supports the Raspbian OS (Operating System).

The architecture includes a WSN, which contains sensors that transmit environmental measurements (temperature, humidity, carbon monoxide, and ultraviolet intensity) to a node (Raspberry Pi 3). With Python programming language and Application Programming Interface (APIs), the information from the sensors is sent to the cloud for statistics, analysis, and visualization.

2) GENERAL STRUCTURE

In the design of the IDS with an anomaly-based approach addressed in the research, a methodology for the development of data science and ML projects has been used, with the following functions as data collection, data preparation, ML model evaluation, anomaly detection, control mechanisms, alarm, and report.

The traffic is captured as a .pcap file for data collection to be exported as Comma Separated Values (CSV). The process is described with the following steps:

1) Initialize the system: The IDS system is initialized to start the data collection process.
2) Configuring and performing a network traffic analysis: The network traffic is analyzed by configuring the necessary parameters for each case.
3) Determine whether or not to tag the traffic: A decision is made on whether to tag the captured traffic for further analysis.
4) Import PCAP file to capture traffic: The captured network traffic is imported as a PCAP file for further processing.
5) Whether or not to save the report as CSV: An option is given to save the generated report in CSV format.

The data preparation step converts the input data into patterns the ML models can process. The data receive a cleaning and removing unnecessary information (new CSV file created). The ML models are trained with the dataset in different scenarios using 80% data for training and 20% for testing. Also, 10-fold cross-validation is used to ensure results. A set of metrics like accuracy, F1-score, recall, and precision supports the testing.

The trained ML models can now detect intrusions or unauthorized access to the network. The IDS displays real-time alarms when an intrusion is detected, and the report is saved in a database to maintain a later visualization register.

B. DATA

The dataset (IDSAI) contains a total of 1,000,000 samples. Initially, it included 24 features. Initial preprocessing (delete features such as IP addresses and ports because they are easily adjustable by attackers) reduces the dataset to 19 variables and the two label columns (1,000,000 × 21). Half of the data are non-intrusion samples; the other 500,000 are intrusions. Each intrusion class includes a total of 50,000 data samples. In total, the database contains ten types of intrusions.

The IDSAI dataset presented here addresses the challenge of data set imbalance in network traffic analysis. To ensure balance and mitigate bias, the dataset has been meticulously designed by capturing an equal number of samples for each intrusion type. This balanced dataset is crucial in overcoming the inherent bias present in imbalanced data, where certain attack types are less common than others and normal network traffic dominates. The IDSAI dataset serves as a valuable resource for training and evaluating machine learning models in network intrusion detection, offering real attacks from diverse sources and a balanced representation of intrusion classes. By providing this balanced dataset, researchers can develop and evaluate machine learning models more effectively, leading to accurate and reliable intrusion detection in real-world scenarios.

1) FEATURES

The features of the proposed dataset are frequently used in other studies related to intrusion detection systems with approaches based on signatures and anomalies. The researchers selected them after studying the entire data and its structure. In the same way, some of them were defined in previous studies by other works [32], [33], [34]. Below are the dataset features' names, the data type, and a brief description.

- *frame_len* (int64): frame length.
- *udp_len* (int64): UDP length, at value 0, indicates that this instance is not of a UDP protocol.
- *ip_ttl* (int64): time to live, the value 0 indicates that this instance is not an IP protocol.
- *delta_time* (float64): Time delta of the captured frame concerning the previous one.
- *icmp_type* (int64): type, the value 19 indicates that this instance is of an invalid ICMP type.
- *tos* (int64): Label the quality of service requested by the IP datagram.
- *ip_flags_rb* (int64): IP flag reserved bit, at value 2 indicates that this instance is unknown, which is not IP protocol.
- *ip_flags_df* (int64): IP flag does not fragment. The value 2 indicates that this instance is unknown.
- *ip_flags_mf* (int64): IP flag plus fragments, in value 2 indicates that this instance is unknown, which is not IP protocol.
- *tcp_flags_res* (int64): TCP reserved flag, in value 2 indicates that this instance is not TCP protocol.

TABLE 2. The ten attacks on the IDSAI and the normal data. It shows the category, subcategory, affected protocols, and attack tool.

Number	Category	Subcategory	Protocols	Attack tool
1	Denial of Service (DoS)	ICMP flood	ICMP	Hping3
2	DoS	SYN/ACK y RST Flooding	TCP	Hping3
3	DoS	SYN Flooding	TCP	Hping3
4	DoS	SYN Flooding faster	TCP	Hping3
5	MiTM	ARP spoofing	ARP, TCP, UDP, ICMP	Ettercap
6	DDos	MAC flood	IPv4, ICMP type packets	Dsniff
7	DoS	Fragmentación IP	IP	Hping3
8	Brute Force	Brute Force on SSH service	TCP	Ncrack
9	UDP port scan	UDP port scan	UDP	Nmap
10	TCP Null	TCP Null	TCP	Nmap
11	Normal	Normal	TCP, UDP, ICMP, IP, ARP	-

- *tcp_flags_ns* (int64): TCP Nonce Flag, at value 2 indicates that this instance is not a TCP protocol.
- *tcp_flags_cwr* (int64): TCP Congestion Window Reduced (CWR) Flag, value 2 indicates that this instance is not a TCP protocol.
- *tcp_flags_ecn* (int64): TCP ECN-Echo Flag, value 0 for inactive, 1 for active, and value 2 indicates that this instance is not a TCP protocol.
- *tcp_flags_urg* (int64): Urgent TCP flag, value 0 for inactive, 1 for active, and value 2 indicates that this instance is not a TCP protocol.
- *tcp_flags_ack* (int64): TCP Acknowledgment flag, value 2 indicates that this instance is not a TCP protocol. It indicates whether the segment carries a valid acknowledgment number.
- *tcp_flags_push* (int64): TCP push flag, value 2 indicates that this instance is not a TCP protocol. It indicates whether it immediately data transferred to the application.
- *tcp_flags_reset* (int64): TCP Reset flag can be of 3 values, 0 for inactive, 1 for active, and 2 if it is not TCP protocol.
- *tcp_flags_syn* (int64): TCP synchronization flag can be of 3 values, 0 for inactive, 1 for active, and 2 if it is not TCP protocol.
- *tcp_flags_fin* (int64): End TCP flag is used in finishing the connection, value 2 if it is not TCP protocol.

2) INTRUSIONS OR CLASSES

In total, the database contains ten types of intrusions called:

1) ICMP echo request Flood / Ping Flood
2) SYN/ACK and RST Flooding
3) SYN/ACK Flooding
4) SYN Flooding faster
5) ARP spoofing
6) DDoS MAC Flood
7) IP Fragmentation
8) Brute Force SSH
9) UDP port scan
10) TCP Null

Table 2 shows the category and subcategory of attacks, the impacted protocols, and the attack tool used. The traffic capture tool used was Wireshark in all cases, and the attacked device was Raspberry Pi in all cases.

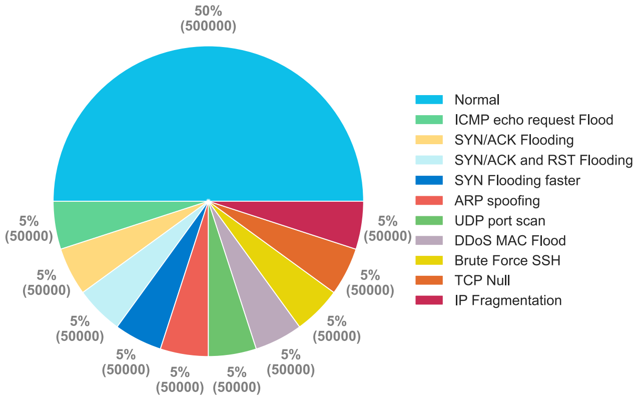


FIGURE 1. Class distribution. The IDSAI dataset is balanced for the ten intrusions and as a binary way of intrusion (joining ten intrusions) and non-intrusion (Normal) data.

Figure 1 shows the data distribution by classes. There are 50,000 data samples for each intrusion (500,000 samples for intrusions) and 500,000 non-intrusion or Normal samples.

Below are the names of the classes or intrusions with a brief description.

- *ICMP echo request Flood / Ping Flood:* the attacker uses a botnet to send large numbers of ICMP packets to the victim host to exhaust available bandwidth and prevent the victim host from being accessible to legitimate users of the system. It generates approximately 5,368 packets per second [35].
- *SYN/ACK and RST Flooding:* this attack causes the victim host to acquire a large volume of fake RST packets not registered in a session started in the victim's database, causing a crash because computational resources are broken when trying to compare the large number of packages received caused total system failure or reduced system performance to a minimum [36].
- *SYN/ACK Flooding:* it is an attack that generates a denial of service or a total collapse of the system due to the excess of SYN + ACK response packets made by a victim host and where the host consumes all memory, CPU and other resources to minimize the attack, but it is impossible to deal with it due to the congestion formed by the response packets. This attack generates approximately 7,500 packets per second [36].
- *SYN Flooding faster:* it is an attack in which the intruder sends a high volume of TCP/SYN packets originating from one or several false addresses to a victim system. The system tries to reply with ACK-SYN packets to a group of false IPs, which will not acknowledge a receipt. Thus leaving a half-open connection that extinguishes the system's memory resources and, consequently, a deterioration in the system's performance or, even worse, a total crash [36], [37].
- *ARP spoofing:* it is a malicious attack that transmits false ARP messages over the local network by linking the MAC address of an intruder with the IP of a

legitimate host, causing the interception, modification and even the denial of network data frame traffic [38].

- *DDoS MAC Flood:* the primary purpose of the MAC flood attack is to delete the MAC table. An intruder connected to a switch port floods many frames onto the Ethernet interface. Using false source MAC addresses, the attacker loads the memory of the switches, which is where they are stored in the MAC table; it causes legitimate users to be removed from the table [39]
- *IP Fragmentation:* attacks are common denial of service attacks. The intruder will try to make a fraudulent implementation of IP fragmentation and confuse the operating system into recomposing the original datagram and thus crash the target system. In addition, the attack intends to modify the information to add inconsistencies once the original datagram has been reconstructed; another harm is flooding the IP stack of the victim host [40].
- *Brute Force SSH:* attack is used to gain unauthorized access to a host, server, or other protected information through illegal access with authentic client names and passwords, which has been achieved with a prediction procedure of the same using all the usernames and passwords of an organization. This type of threat could be prevented through the intrusion detection, and prevention mechanism (IDS/IPS) that controls the number of access attempts [41].
- *UDP port scan:* is a prevalent security threat used by intruders on a victim to find open doors, learn the operating system and services that allow illegitimate logins by sending and receiving packets to specific ports on a host, and finding faults In the system, in this way, they monitor the response of a network host and inquire about the status of a port, and the ease of access [37].
- *TCP Null:* attack, the victim receives TCP packets that come with null values in the flag area of the TCP header, and it is because none of the six TCP flags (URG, ACK, PSH, RST, SYN, FIN) have been set. If the port is enabled on the victim, NULL packets are unknown. Instead, the attacker would receive an RST packet. This vulnerability scans the victim's ports and builds a large attack [42].

3) BOT-IoT DATASET FOR TESTING

The unbiased evaluation of the models is applied using the Bot-IoT dataset [18] created by the University of Canberra consisting of regular and botnet traffic; the authors included the original Pcap files, Argus, and CSV. The researchers labeled the dataset and classified it by class. Part of the raw network package files (Pcap) from the BoT-IoT dataset was used for the experimentation. By building a script, 19 features were extracted. The new distribution is 200,000 instances of the normal class and five attack classes, such as DDoS_TCP, DDoS_UDP, DoS_TCP, DoS_UDP, Reconnaissance_OS_Fingerprint, and Reconnaissance_Service_Scan, with 20,000 samples in each category. The dataset is unbalanced, with the DDoS attack with 1,926,624 records, DoS with 1,650,260, Reconnaissance with 91,082, and Normal with 477. Many researchers have used 5% of the total data, corresponding to 1.07 GB of the full size.

- *DDoS_TCP*: consists of the increase or flooding with a large volume of malicious packets through botnets directed at a victim to exhaust the computational resource or absorb the bandwidth. Because the attack can spread across multiple machines, it will be challenging to differentiate between legitimate users and intruders [43], [44].
- *DDoS_UDP*: This attack is performed by flooding User Datagram Protocol packets. The intruder floods a random port on the device with UDP packets forcing the victim to check the affected port constantly. However, it is being used or listened to by the system. The affected devices send massive messages of no access and ICMP error. This action depletes the victim's resources, causing the unavailability of the system to legitimate users [36].
- *DoS_TCP*: This is a typical DoS attack, where an attacker dispatches TCP connection requests to clog existing ports on the system, making it impossible to accept real connections from authenticated users [45].
- *DoS_UDP*: sends many corrupted UDP packets to exposed ports of a victim host (UDP does not need a communication link like TCP). When a UDP packet is received on a specific host port, it is determined if any application is active or if the host sends an ICMP target unreachable message to the replaced source address. It is clear that if a host receives a large number of UDP packets, the system's performance suffers until it becomes unavailable [37], [44].
- *Reconnaissance_OS_Fingerprint*: is a technique used in ethical hacking that allows identifying the operating system that runs on a remote host susceptible to attack and what vulnerabilities the host systems present that facilitate the subsequent phase of an attack [42], [46].
- *Reconnaissance_Service_Scan*: the attack consists of an address analysis to find out the weaknesses of the active services in a network of hosts. Intruders often perform address study in the first phase, then carry out cyberattacks such as DoS and progress to devastating attacks such as DDoS attacks [46].

C. MODELS

In this work, a total of 8 ML algorithms were evaluated, such as XGB [47], GB [48], DT [49], RF [50], ET [51], LR [52], GNB [53] and LDA [54]. The totrp 5 ML algorithms with the best performance for intrusion detection are briefly explained below.

- *XGB* is an ensemble technique developed based on Gradient Boosting. It is trained using a simultaneous set of regression trees, whose result is the sum of the score of each tree [47]. Reference [55] added

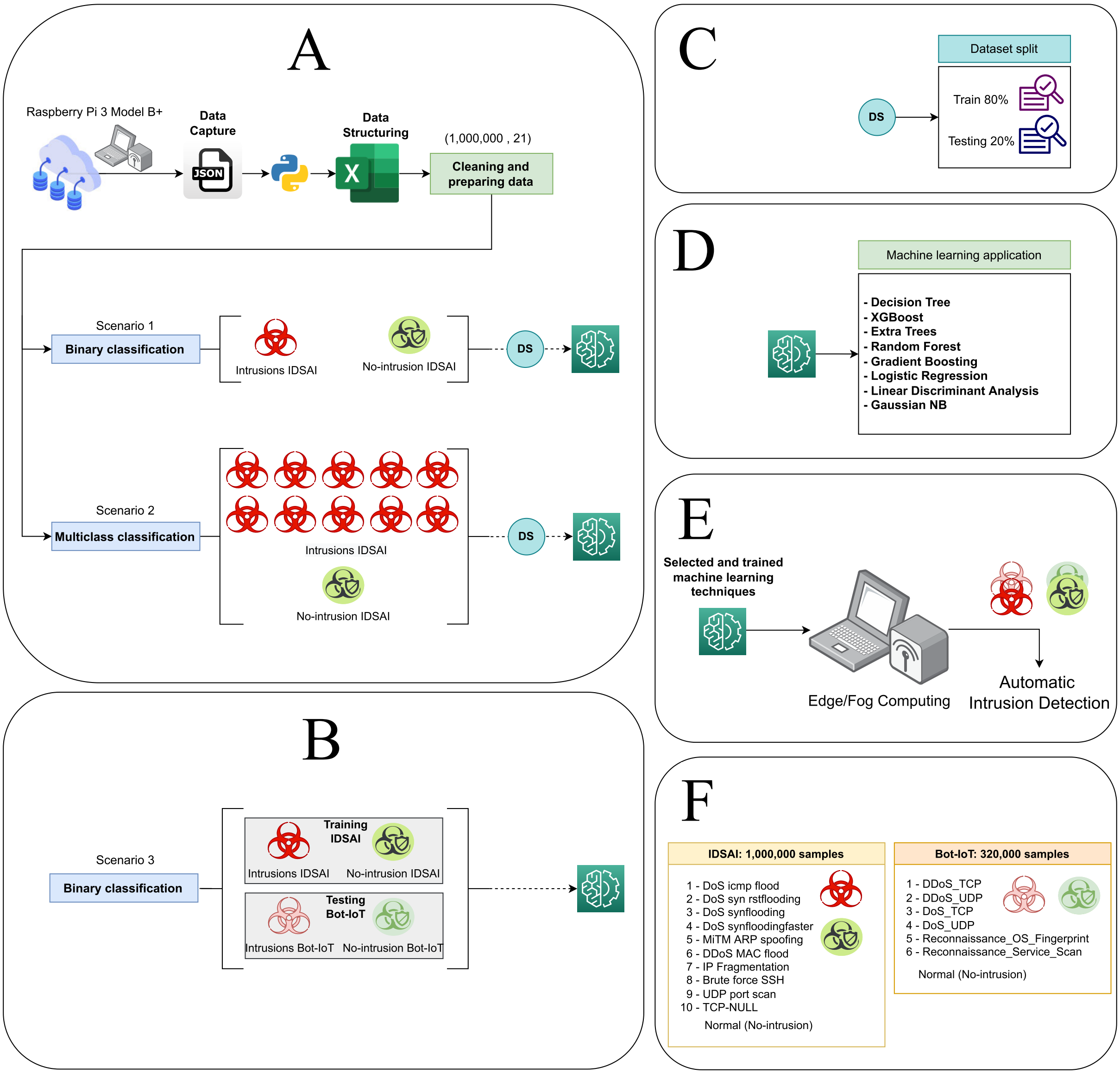


FIGURE 2. Pipeline of the overall process of the proposed methodology. (A) corresponds to the experimentation in IDSAI data for binary and multiclass classification. (B) corresponds to the external validation using the Bot-IoT dataset. (C) refers to the Dataset Split (DS) process. (D) ML application. (E) ML models applied in the IDS. (F) dataset proposed in this work IDSAI and external dataset for validation (Bot-IoT).

some improvements to it in 2016 and named it XGB. This algorithm combines the idea of Boosting, overcoming the speed and accuracy of limited calculations and blocks, and simultaneously orders each function. It allows parallelizing the computation when searching for the best-split point, which significantly accelerates the calculation speed [50].

- *GB* is used for solving regression and classification problems. It is equivalent to the Ada Boost algorithm, with a mixture of weak classification models generally developing a DT model. Reference [48] explains that the general idea is to prepare sequentially, each of which attempts to correct its predecessor.
- *DT* is used to solve classification and regression problems. According to a data set, it builds diagrams of logical structures with which it represents and categorizes a series of conditions given consecutively to solve a problem [49]. It comprises a tree scheme with trees and decision nodes (the result of the decision).
- *RF* this algorithm trains many DTs, each using a random subset of samples and features. RF achieves increased tree diversity and gives better outcomes [50].
- *ET* are DT ensembles. This approach adds randomization to the model training process by employing random decision thresholds for each feature rather than seeking the best feasible [51].

D. MODEL TRAINING

The proposed experimentation process is divided into a total of 3 scenarios (see **Figure 2**). The first two scenarios correspond to the experimentation for the proposed IDSAI dataset (see **Figure 2 (A)**). In scenario 1, binary classification is performed, predicting whether there is an intrusion. In scenario 2, multiclass intrusion identification is proposed; in this case, it will be sought to say, given that there is an intrusion, what it could be, and with what certainty.

Scenario 3 seeks to perform an external validation of the dataset proposed for intrusion detection (see **Figure 2 (B)**). The training data is the IDSAI dataset presented in this work, and the testing data is the Bot-IoT dataset. To test Bot-IoT data, the network traffic capture was structured to the same features with which the models were trained using the IDSAI dataset. With this set of experiments, it is hoped to verify that the models work with different datasets and even have different types of intrusions.

Experiments for scenarios 1 and 2 are performed using ML with Hold-Out by splitting data with 80% for training and the remaining for testing (see **Figure 2 (C)**). A total of 8 ML algorithms were evaluated in all scenarios, and the best five were selected (see **Figure 2 (D)**). The best ML models are saved and sent to the IDS environment, which performs the predictions in real-time directly on a device (see **Figure 2 (E)**).

The Bot-IoT database (see [18] for more detailed information) has data without intrusions (200, 000 samples) and five intrusions such as DDoS TCP, DDoS UDP, DoS TCP, DoS UDP, Reconnaissance OS Fingerprint, and Reconnaissance Service Scan, with 20, 000 samples, each one (see **Figure 2 (F)**). In scenarios 1 and 2, the IDSAI dataset proposed in this research is used. Furthermore, in scenario 3, predictions are made about Bot-Iot to verify the effectiveness of training ML algorithms using IDSAI data.

1) HYPERPARAMETER TUNING

To perform the hyperparameter tuning (see **Figure 3**) for ML models in this work are completed the following steps: selection of a set of hyperparameters, establishment of the accuracy as the metric, use only training data to select hyperparameters, according to the grid of hyperparameters and using Grid Search tool the models are trained using 3-fold cross-validation to optimizing the hyperparameter settings. This exhaustive search for the best hyperparameter values for the ML models is applied to all scenarios. After hyperparameter optimization, were chosen the best ones to train the best ML algorithms and predict on testing data (never seen in tuning or training). The hyperparameters of the top-performing ML algorithms in each classification scenario are showcased in **Table 3**. Detailed explanations of these hyperparameters can be found on the scikit-learn website [56], [57], providing a comprehensive understanding of their functionality.

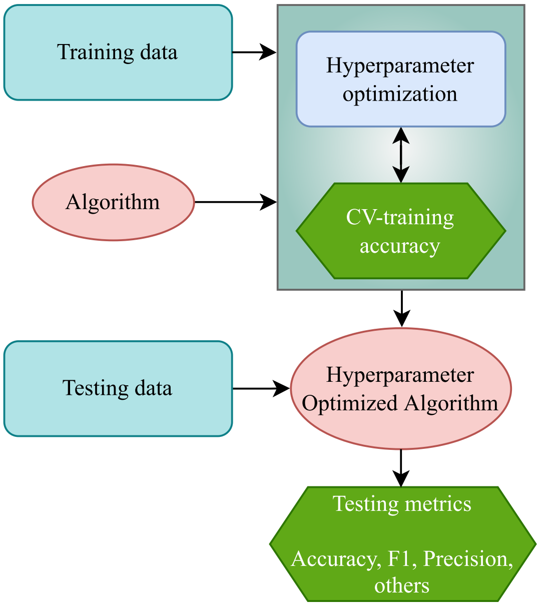

Training data
Hyperparameter optimization
Algorithm
CV-training accuracy
Testing data
Hyperparameter Optimized Algorithm
Testing metrics
Accuracy, F1, Precision, others

FIGURE 3. Schematic diagram of the hyperparameter tuning process. It also shows metrics for testing data.

E. PERFORMANCE ASSESSMENT

This research utilizes eight metrics, which are detailed in [58], [59], and [60]. The four measures used to calculate these metrics are: True Positive (TP) for correctly predicted intrusions, True Negative (TN) for correctly predicted non-intrusions, False Positive (FP) for incorrectly predicted intrusions, and False Negative (FN) for incorrectly predicted non-intrusions. For evaluation, this study considers measures such as accuracy, precision, recall, F1-score, ROC curves, the Area Under the ROC Curve (AUC ROC), cross-validation and execution time.

1) ACCURACY

Accuracy is a metric used to evaluate how well a classification model performs in making predictions. This is done by dividing the total number of correct predictions made by the model with the total number of predictions made [60]. **Equation (1)** represents the percentage of instances correctly classified out of the total [58], [59]:

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{1}$$

2) PRECISION

The precision metric measures the proportion of positive cases that are correctly identified by a model among all the cases identified as positive, including true positives and false positives. **Equation (2)** shows that precision is calculated as the number of true positives divided by the sum of true positives and false positives [58], [59], [60].

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{2}$$

TABLE 3. Hyperparameters of the best ML algorithms for each classification scenario.

Scenario	Algorithm	Hyperparameter name	Hyperparameter value
1	XGB	learning_rate	0.3
		n_estimators	100
		max_depth	6
		min_child_weight	1
		gamma	0
		subsample	1
		colsample_bytree	0.75
		seed	0
		eval_metric	'mlogloss'
1	GB	loss	'log_loss'
		learning_rate	0.1
		min_samples_split	2
		min_samples_leaf	1
		min_weight_fraction_leaf	0.0
		max_depth	9
		criterion	'friedman_mse'
		random_state	64
		subsample	1.0
		n_estimators	100
2	XGB	learning_rate	0.1
		n_estimators	100
		max_depth	10
		min_child_weight	1
		gamma	0
		subsample	1
		colsample_bytree	1
		seed	0
		eval_metric	'mlogloss'
2	RF	max_depth	20
		n_estimators	120
		criterion	'gini'
		random_state	64
3	XGB	learning_rate	0.1
		n_estimators	100
		max_depth	10
		min_child_weight	1
		gamma	0
		subsample	1
		colsample_bytree	1
		seed	0
		eval_metric	'mlogloss'
3	GB	loss	'log_loss'
		learning_rate	0.1
		min_samples_split	2
		min_samples_leaf	1
		min_weight_fraction_leaf	0.0
		max_depth	3
		criterion	'friedman_mse'
		random_state	8
		subsample	1.0
		n_estimators	100

3) RECALL

It is known as the true positive rate [58], [59], [60]. It is the percentage of positive cases correctly detected by the model (see **Equation 3**).

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{3}$$

4) F1

F1 is a metric used to evaluate the model's ability to accurately identify both positive and negative cases, especially when the data is imbalanced and the positive class is rare [58], [59]. It is calculated as the harmonic mean of Precision and Recall, and is particularly useful for uneven classes [60]. **Equation 4** can be used to estimate the model's average precision.

$$\text{F1} = 2\text{x}\frac{\text{Precision x Recall}}{\text{Precision} + \text{Recall}} \tag{4}$$

5) AUC ROC

The ROC curve shows the relationship between true positive rate (TPR) and false positive rate (FPR) at different decision thresholds, useful for comparing classification models and finding the best threshold. AUC is a numerical measure summarizing the model's overall performance, with values closer to 1 indicating better performance. A model with high sensitivity and specificity is represented by an ideal curve that reaches the upper-left corner, and AUC ROC is associated with this curve [58], [59], [60].

6) CONFUSION MATRIX

The confusion matrix is a table that summarizes the relationship between a model's predictions and the true labels of the data. It includes TP, TN, FP, and FN. This matrix is useful for visualizing a model's performance in terms of its successes and errors. Each row of the matrix corresponds to the actual class, while each column represents the number of predictions made for each class. It also helps to identify when one class is confused with another [58], [59], [60].

7) CROSS VALIDATION (CV)

A 10-fold CV is used in all experiments [58], [59]. It consists of repeating and calculating the arithmetic mean obtained from the evaluation measures on different partitions. It ensures the results are independent of the training and validation data split. It is regularly used in environments where the main objective is to predict and estimate the accuracy of a model to be put into production.

8) RUN TIME

(RT) Indicates the time for an ML model to train and predict [58], [59].

F. FEATURE IMPORTANCE

The feature importance is calculated using supervised ML algorithms for binary and multiclass classification. Some feature importance methods are embedded into the scikit-learn software for multiple ML models (*feature_importances_* and *coef_* properties). The DT algorithm, for example, has the *feature_importances_* property. The *feature_importances_* is accessible in decision tree models and tree ensembles and reflects how much this feature is utilized in each tree. The coefficients with the most significant values are relevant since they lend more weight to the predictions [56].

The Yellowbrick tool gets the feature importance from the models (in the second plane, it utilizes *feature_importances_* and *coef_*). This tool also can stack feature importances for top and bottom importance [61]. It enables us to learn and know which factors have the most influence on each scenario.

TABLE 4. Scenario 1 and 2, ML application on IDSAI dataset. Results for testing data in binary and multiclass classification.

Scenario	Algorithm	Accuracy [%]	F1 [%]	Recall [%]	Precision [%]	ROC AUC [%]	CV [%]	Training time [sec]	Testing time [sec]	CV time [sec]
1 - Binary classification	XGB	94.97	94.97	94.97	95.28	99.09	94.98 ± 0.05	21.6598	0.0828	131.7631
	GB	94.99	94.98	94.99	95.28	99.09	94.97 ± 0.06	369.9567	1.4571	1,263.8644
	RF	94.98	94.97	94.97	95.27	99.08	94.96 ± 0.05	48.6616	0.4294	608.5278
	DT	94.95	94.95	94.95	95.25	98.98	94.95 ± 0.05	3.8859	0.0221	10.5412
	ET	94.68	94.68	94.68	94.92	98.51	94.65 ± 0.07	52.2130	0.3192	681.4079
2 - Multiclass classification	XGB	92.64	91.89	92.64	92.59	99.47	92.61 ± 0.06	164.0557	0.8564	1,133.7668
	RF	92.60	91.92	92.60	92.45	99.46	92.56 ± 0.06	45.1873	0.6859	638.7699
	DT	92.56	91.89	92.56	92.37	99.34	92.51 ± 0.06	7.1727	0.0317	16.5630
	GB	92.49	91.65	92.49	92.51	99.44	92.44 ± 0.08	1,006.6022	4.4309	4,770.9118
	ET	92.24	91.69	92.24	91.88	99.02	92.20 ± 0.06	84.3063	0.9819	1,115.7969

This research uses the DT algorithm since it is very efficient in training and prediction times while maintaining a good detection ability. It also gets good results by using features efficiently.

G. RESOURCES

Python 3.8 is used to develop and execute the algorithms presented in this study. The computer runs Windows 11 (64-bits), and it has an Intel(R) Core(TM) i9-10980HK CPU @ 2.40 GHz 3.10 GHz processor, 32 GB of RAM, and an NVIDIA GeForce RTX 2070 Super GPU (8GB). The code and data are available in https://github.com/BioAITeam/Intrusion-Detection-System-using-Machine-Learning.

IV. EXPERIMENTAL RESULTS AND ANALYSIS

A total of three possible scenarios have been defined that cover all perspectives of intrusion classification using supervised ML. The first two scenarios are based on the proposed IDSAI dataset. The last is for external validation using the Bot-IoT dataset. The experimentation results are framed in binary and multiclass classification for the IDSAI dataset. The results include tables covering many metrics to measure the performance, such as Accuracy, F1-score, Recall, Precision, ROC AUC, CV, and Times. In addition, this work presents confusion matrices and ROC Curves with the AUC and Confidence Intervals.

A. IDSAI DATASET FOR INTRUSION DETECTION

Table 4 shows the results obtained with the best five ML models for scenarios 1 and 2. The algorithms are ordered from the highest to the lowest CV value (recommended metric before using ML models in production).

The performance of the classifier was evaluated using ROC curves with confidence intervals, as illustrated in **Figure 4**. The area under the ROC curve served as a metric to measure the classifier's ability to discriminate between classes. In a similar manner, ROC curves with confidence intervals were generated for multiclass classification (scenario 2), as presented in **Figure 5**. These curves provide valuable insights into the classifier's performance in distinguishing between different classes. The results of this study highlight the effectiveness of the XGBoost classifier in both binary and multiclass classification tasks on the IDSAI dataset.

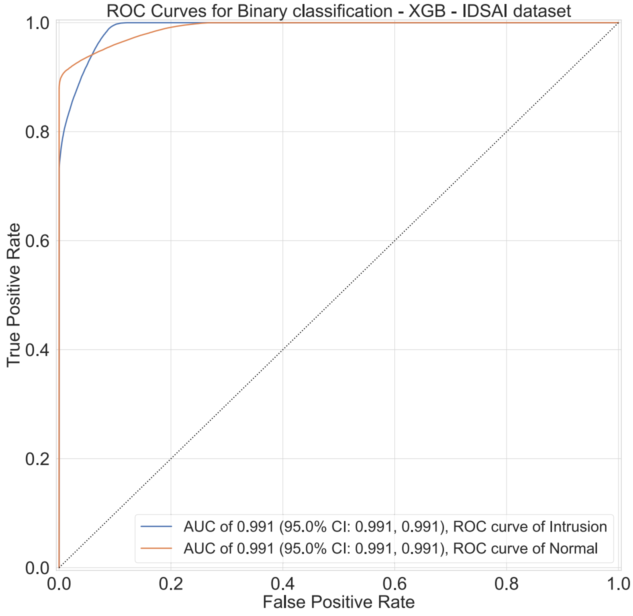


FIGURE 4. ROC curves with CI for binary classification on the IDSAI dataset using 20% of the data for testing (scenario 1).

For scenario 1, which corresponds to a binary classification (intrusion, non-intrusion), the best algorithm is XGB, obtaining an accuracy of 94.97%. Regarding training time, the best algorithm is DT, needing only 3.8859 seconds while maintaining an accuracy of over 94%. The algorithm that had the worst times is GB needing 369.9567 seconds; due to this, GB is not so desirable in a production environment even having good performance (94.97 ± 0.06).

In the case of scenario 2 for multiclass classification (non-intrusion, ICMP echo request Flood / Ping Flood, SYN/ACK & RST Flooding, SYN/ACK Flooding, SYN Flooding faster, ARP spoofing, DDoS MAC Flood, IP Fragmentation, Brute Force SSH, UDP port scan, TCP Null), once again, the XGB algorithm has the best accuracy (92.64%). Once again, DT is the algorithm with the best performance (92.51 ± 0.06), requiring less training time (7.1727 seconds). The GB algorithm, which in scenario 1 was second, is now fourth and the most inefficient in time, needing 1,006.6022 seconds to train.

Figure 6 shows the confusion matrix for the identification of ten intrusions and non-intrusion data (scenario 2).

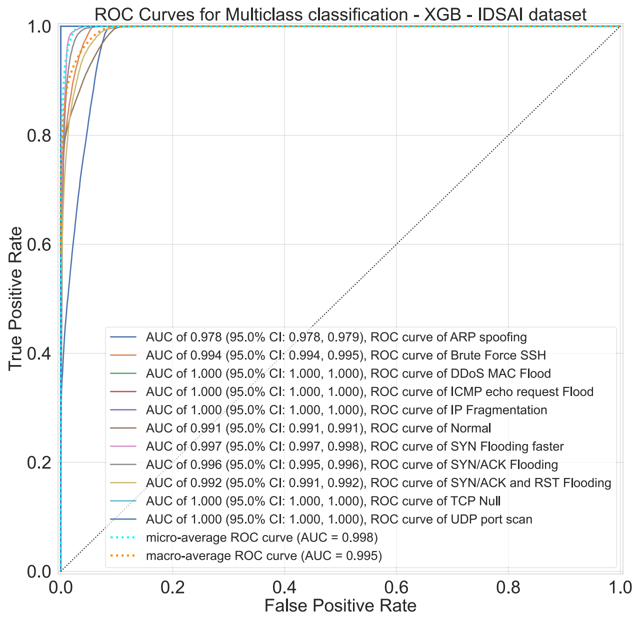
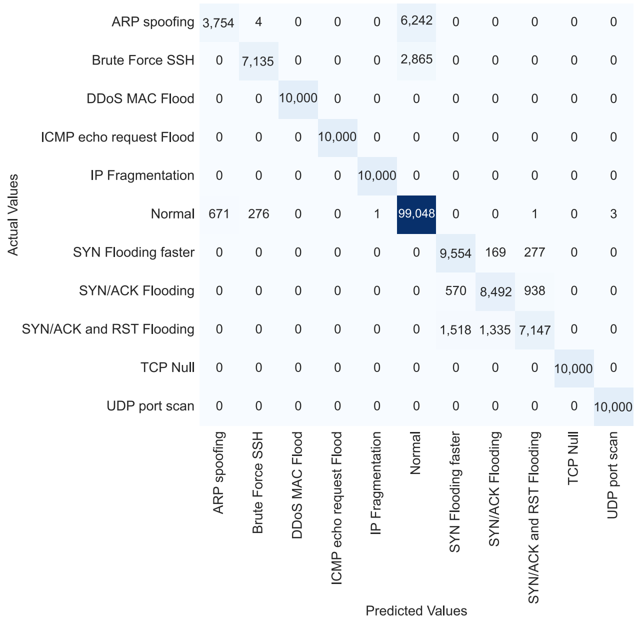


FIGURE 5. ROC curves with CI for multiclass classification on the IDSAI dataset using 20% of the data for testing (scenario 2).



	ARP spoofing	Brute Force SSH	DDoS MAC Flood	ICMP echo request Flood	IP Fragmentation	Normal	SYN Flooding faster	SYN/ACK Flooding	SYN/ACK and RST Flooding	TCP Null	UDP port scan
ARP spoofing	3,754	4	0	0	0	6,242	0	0	0	0	0
Brute Force SSH	0	7,135	0	0	0	2,865	0	0	0	0	0
DDoS MAC Flood	0	0	10,000	0	0	0	0	0	0	0	0
ICMP echo request Flood	0	0	0	10,000	0	0	0	0	0	0	0
IP Fragmentation	0	0	0	0	10,000	0	0	0	0	0	0
Normal	671	276	0	0	1	99,048	0	0	1	0	3
SYN Flooding faster	0	0	0	0	0	0	9,554	169	277	0	0
SYN/ACK Flooding	0	0	0	0	0	0	570	8,492	938	0	0
SYN/ACK and RST Flooding	0	0	0	0	0	0	1,518	1,335	7,147	0	0
TCP Null	0	0	0	0	0	0	0	0	0	10,000	0
UDP port scan	0	0	0	0	0	0	0	0	0	0	10,000

FIGURE 6. Confusion matrix for multiclass classification (scenario 2), ten different intrusions and normal data.

According to the **Figure 6**, the models will likely get confused and predict SYN/ACK Flooding or SYN/ACK & RST Flooding when the attack is SYN Flooding faster. On the other hand, it could also be predicted as SYN Flooding faster or SYN/ACK Flooding when the intrusion is SYN/ACK & RST Flooding. It also tends to identify ARP spoofing and Brute Force SSH attacks as non-intrusions data. Therefore, these two attacks are more complex to detect and confuse as non-intrusions.

B. FEATURE IMPORTANCE

ML algorithms tolerate the complexity and heterogeneity of data structures, making it possible to find underlying patterns in the data. **Figure 7** graphically shows features' relative importance for binary and multiclass classification, scenarios 1 and 2. This analysis is performed using the IDSAI dataset. The SHAP (SHapley Additive exPlanations) [62] method was used to analyze the relative importance of features in a binary classification scenario using the IDSAI dataset. The results, shown in **Figure 8**, revealed that Feature *frame_len* had the highest importance in scenario 1. Furthermore, it was observed that the results obtained with SHAP were similar to those obtained using other interpretability techniques (see **Figure 7**). This consistency strengthens confidence in the results and provides valuable insights for feature selection and data analysis.

This work (see **Figure 7 (A)**) shows that the binary classification can be done using the following 11 features (where only the first three have each one a relative importance greater than 20%.):

1) *ip_ttl*
2) *frame_len*
3) *tos*
4) *tcp_flags_ack*
5) *udp_len*
6) *ip_flags_df*
7) *delta_time*
8) *icmp_type*
9) *tcp_flags_syn*
10) *tcp_flags_fin*
11) *tcp_flags_push*

Figure 7 (B) shows that multiclass classification can be done using the following 12 features (where only the first six features have each one relative importance greater than 20%.):

1) *frame_len*
2) *ip_ttl*
3) *tos*
4) *delta_time*
5) *tcp_flags_push*
6) *icmp_type*
7) *ip_flags_mf*
8) *ip_flags_df*
9) *udp_len*
10) *tcp_flags_reset*
11) *tcp_flags_fin*
12) *tcp_flags_syn*

C. VALIDATION USING BOT-IoT DATASET

One of the most critical problems in model training is the generalization in front of data of different natures. Reference [63] proposed an alternative approach to assess the generalizability of a model, using two distinct but related datasets instead of a single one. The first dataset is used

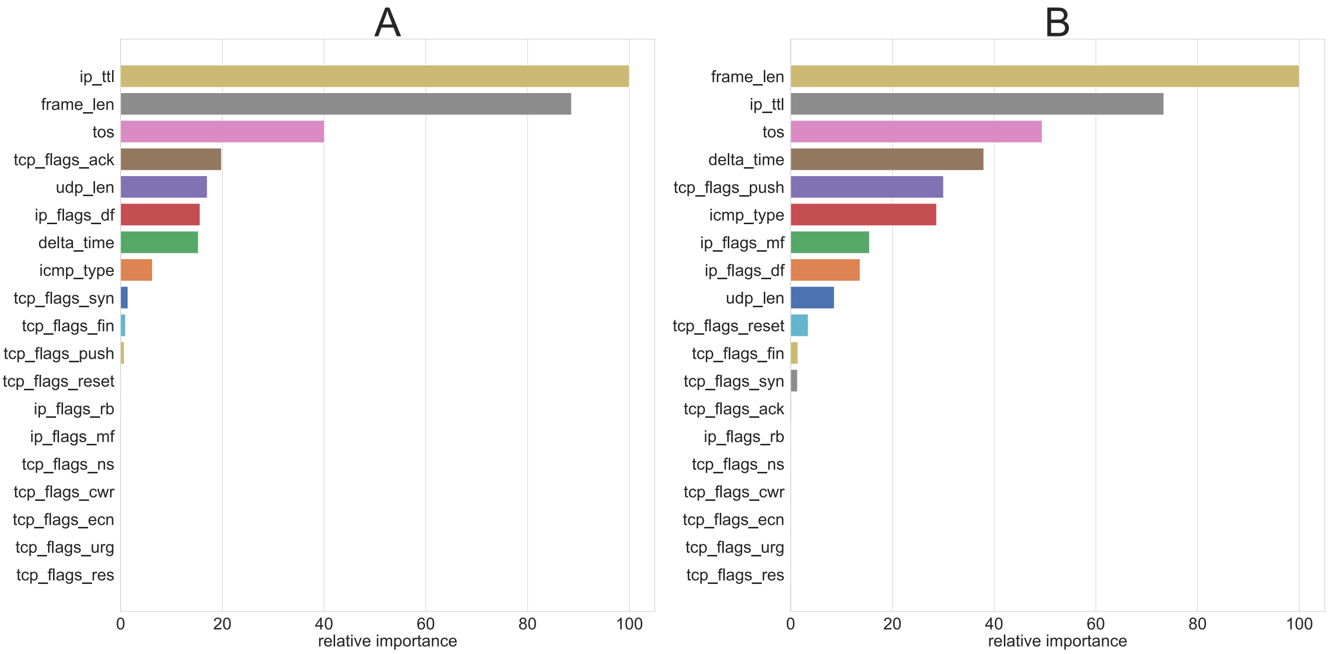


FIGURE 7. Feature importance on IDSAI dataset for intrusion detection using DT algorithm. (A) Scenario 1, binary classification. (B) Scenario 2, multiclass classification.

TABLE 5. Scenario 3 shows an external validation with results for predictions on the Bot-IoT dataset using ML models trained using the IDSAI dataset.

Scenario	Algorithm	Accuracy [%]	F1 [%]	Recall [%]	Precision [%]	ROC AUC [%]	Training time [sec]	Testing time [sec]
3 - Binary classification	XGB	94.80	94.83	94.80	94.97	97.50	25.8387	0.1633
	GB	90.59	90.71	90.59	91.69	96.55	70.2479	0.2784
	DT	90.56	90.68	90.56	91.57	91.68	13.1012	0.0229
	RF	90.56	90.68	90.56	91.64	95.51	51.1536	0.4067
	ET	89.74	89.88	89.74	91.06	95.17	129.8331	0.2858

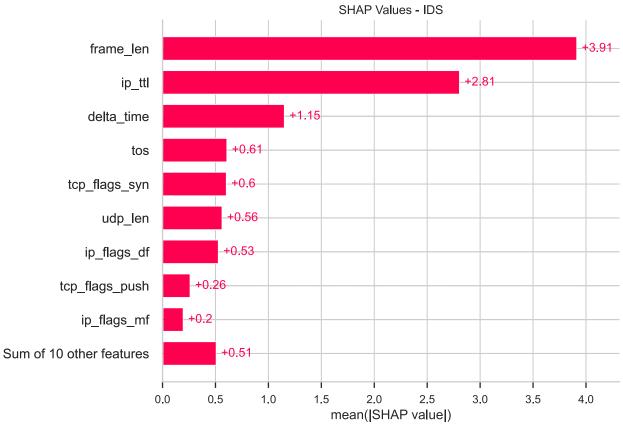


FIGURE 8. Feature importance using SHAP on IDSAI dataset for intrusion detection using XGB algorithm on Scenario 1.

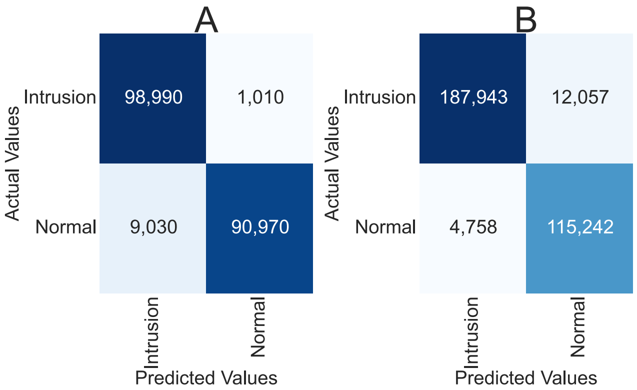


FIGURE 9. Confusion matrices. (A) scenario 1, testing data from the IDSAI dataset. (B) scenario 3, the Bot-IoT as testing data.

for training and validation, and the second is for unbiased evaluation, as developed in this study.

Table 5 shows the results on the Bot-IoT dataset when the algorithms are trained on the IDSAI dataset. The XGB algorithm maintains an accuracy of over 94%, again being the best for intrusion detection. The GB, DT, and RF algorithms achieve accuracies over 90%. ET obtained an accuracy of 89.74%, with the lowest performance of the five selected algorithms. The results are essential, considering that the Bot-IoT dataset does not have the same intrusions as IDSAI. Bot-IoT dataset intrusions are DDoS TCP, DDoS UDP, DoS TCP, DoS UDP, Reconnaissance OS Fingerprint, and Reconnaissance Service Scan. The results show that the algorithms have proper generalization when trained on the IDSAI data.

TABLE 6. Advantages and disadvantages of different works in intrusion detection.

Author	Advantages	Disadvantages
[17]	Proposes the reduction of false alarms in intrusion detection systems using data mining and machine learning techniques, achieving high precision (96.55%) and recall (93.67%) with the KNN algorithm.	Limited to anomaly-based intrusion detection and requires significant computational resources for its implementation.
[20]	Evaluation of various machine learning classifiers, highlighting Random Forest for its precision (87%), accuracy (98%), and F-measure (84%).	Selective feature inclusion for IDS is required and more recent data is needed to detect new or sophisticated intrusions.
[22]	Performance evaluation of ML-based IDS, highlighting K-NN-AIDS, DT-AIDS, and NB-AIDS models in attack detection and minimization of false positives and negatives.	Limitations in detecting new types of attacks with multiple classification and lack of fairer evaluation metrics.
[23]	Application of Python for feature selection and training of machine learning models, highlighting the Extra Tree Classifier for its high stability and accuracy.	Need to use more supervised learning models for testing and to apply clustering in unsupervised learning to reduce overfitting and false positives.
[24]	Proposes a framework and a hybrid algorithm for the selection of machine learning algorithms for the identification of anomalies and intrusions in IoT networks, highlighting Naïve Bayes.	Limitations due to the use of a single dataset and a limited series of machine learning algorithms.
[26]	Creation of CUPID, a new source of intrusion data based on realistic network traffic to improve the effectiveness of ML-based IDS.	Difficulties in obtaining unencrypted real data and scarcity of datasets with specific OT protocols or hybrid networks, in addition to the time and resources required for the participation of live pentesters.
[27]	Proposes ML-based IDS for the detection of injection attacks with high accuracy (99%) using only 8 relevant features, suitable for IoT devices with limited capabilities.	Limitations in the selection of features based on deep learning and need to improve accuracy, considering the time and number of iterations required to reach convergence.
[28]	Design of NIDS to detect multiple types of attacks in IoT networks using various integrated techniques and methodologies.	IDS accuracy may be affected by false positive rates and detection, and the presence of irrelevant and redundant features in network data.
Our work	Use of a new dataset (IDSAI) obtained in a real and balanced attack environment, and application of eight machine learning algorithms to evaluate their effectiveness in intrusion detection. The algorithms achieved up to 94% accuracy in binary predictions (intrusion and non-intrusion) and up to 92% accuracy in multiclass predictions (ten different intrusions and non-intrusions). Similarly, by using models trained with the IDSAI dataset, up to 90% accuracy is achieved in the prediction on the Bot-IoT dataset.	The computational cost and training time can be high depending on the availability of hardware.

Figure 9 shows confusion matrices of scenarios 1 and 3 corresponding to binary classification. The results achieved in binary form are about an accuracy of 94% (see **Figure 9 (A)**). The confusion matrix shows that intrusions into the Bot-IoT dataset are being correctly classified (see **Figure 9 (B)**). The XGB algorithm is used, which is the best in all cases.

Additionally, this paper presents the confusion matrix for a 20% testing data extracted from the Bot-IoT dataset (the confusion matrix for multiclass classification for the Bot-IoT dataset is included in **Figure 10**). The training process was conducted using the remaining 80% of Bot-IoT data, which involved employing all ML models and fine-tuning their hyperparameters. The XGB model was chosen as the best-performing one. The results reveal that although the Bot-IoT dataset is challenging to classify by classes, binary prediction achieves a high level of accuracy, as shown in scenario 3. Regarding multiclass classification, the following are the performance metrics: 26.15 seconds for training time, 0.11 seconds for prediction time, 84.14% for accuracy score, 84.05% for f1 score, 84.14% for recall score, 85.41% for precision score, 97.64% for ROC AUC, and 0.5180 for MSE. Furthermore, cross-validation was conducted, which took 152.44 seconds and achieved an accuracy score of 84.19% with a standard deviation of 0.16%.

The advent of the Internet of Things and its incorporation into smart cities, while advantageous, has ushered in an era of new security and privacy complications, making IoT networks a preferred target for malefactors [1]. These networks' susceptibility is heightened by the inherent constraints of IoT devices such as limited storage, processing capabilities, and memory, as well as the utilization of insecure wireless communications [3], [4]. Existing security methods struggle with these evolving threats due to issues like complexity, resource usage, data quality, and service disruptions [5], [6], [7], [8]. This research contributes to the realm of IoT network cybersecurity by proposing an innovative Intrusion Detection System, demonstrating the power of Artificial Intelligence in intrusion detection, and providing an understanding of the intrinsic causes of different attacks. By doing so, we not only enhance the comprehension of IoT security challenges but

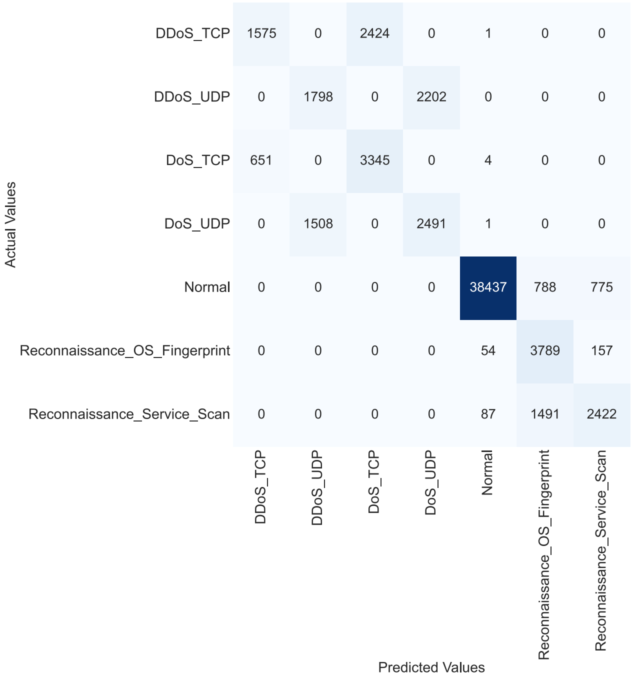


FIGURE 10. Confusion matrix for multiclass classification on the Bot-IoT dataset using 20% of the data for testing.

also contribute to fortifying IoT networks against emerging threats.

D. LIMITATIONS OF THE STUDY

A possible limitation of the current work stems from the nature of the data since attackers are constantly looking to design new ways to attack. Although the IDSAI dataset allowed generalization for prediction on the Bot-IoT dataset, updating the database with new intrusions is advisable to make the ML algorithms more robust.

Table 6 shows a comparison of advantages and disadvantages for various works related to intrusion detection systems. Each author's approach is summarized in terms of the advantages it offers and the corresponding disadvantages or limitations.

V. CONCLUSION

With the growth of the IoT ecosystem, the cybersecurity attack surface has increased. This work presents a sustainable intrusion detection system through supervised ML algorithms. Performance was evaluated using many metrics such as Accuracy, Precision, Recall, F1-score, ROC Curves, ROC AUC, Confusion Matrix, Cross Validation, and Times.

A new dataset (IDSAI) is presented with 1, 000, 000 data samples and 19 features, with intrusions generated in real attack environments. IDSAI dataset has data without intrusions and a total of ten intrusions: ARP spoofing, Brute Force SSH, DDoS MAC Flood, ICMP echo request Flood, IP Fragmentation, SYN Flooding faster, SYN/ACK Flooding, SYN/ACK & RST Flooding, TCP Null, and UDP port scan. IDSAI is a balanced data set with equal number of attacks for each category.

The best ML algorithms for intrusion detection are XGBoost, Gradient Boosting, Decision Tree, Random Forest, and Extra Trees. These ML algorithms can predict the ten specific intrusions achieving an accuracy of over 92%, and in a binary way (intrusion and non-intrusion), achieving an accuracy of over 94%.

On the Bot-IoT dataset, an accuracy of over 90% is obtained. It shows that the models correctly learn to detect intrusions once trained in IDSAI dataset.

In the feature importance analysis for binary and multiclass classification, the following features are found as relevant for intrusion detection: *ip_ttl*, *frame_len*, *tos*, *tcp_flags_ack*, *udp_len*, *ip_flags_df*, *delta_time*, *icmp_type*, *tcp_flags_syn*, *tcp_flags_fin*, *tcp_flags_push*, *ip_flags_mf*, and *tcp_flags_reset*.

In future work, it is recommended to prioritize research on developing a novel intrusion detection system that leverages unsupervised machine learning and anomaly detection techniques. The aim is to compare and evaluate the performance of these techniques using internal and external metrics, with a focus on achieving better performance results while optimizing computational resource consumption. Additionally, efforts can be directed towards refining the IDSAI dataset for model comparison, which involves categorizing new intrusions and their variations to enhance the dataset's effectiveness in evaluating and comparing intrusion detection models.

ACKNOWLEDGMENT

The authors would like to express their gratitude to Universidad Cooperativa de Colombia, Universidad Autónoma de Manizales, Universidad Carlos III de Madrid, Universidad Autónoma de Bucaramanga, and Universidad Veracruzana for their support in this project.

(Gutierrez-Portela Fernando and Arteaga-Arteaga Harold Brayan contributed equally to this work.)

REFERENCES

[1] A. Mourad, H. Tout, O. A. Wahab, H. Otrok, and T. Dbouk, "Ad hoc vehicular fog enabling cooperative low-latency intrusion detection," *IEEE Internet Things J.*, vol. 8, no. 2, pp. 829–843, Jan. 2021, doi: 10.1109/JIOT.2020.3008488.

[2] IoT Analytics. (2022). *IoT Analytics-Your Global IoT Market Research Partner*. [Online]. Available: https://iot-analytics.com/

[3] I. Butun, S. D. Morgera, and R. Sankar, "A survey of intrusion detection systems in wireless sensor networks," *IEEE Commun. Surveys Tuts.*, vol. 16, no. 1, pp. 266–282, 1st Quart., 2014, doi: 10.1109/SURV.2013.050113.00191.

[4] Z. A. Khan and P. Herrmann, "Recent advancements in intrusion detection systems for the Internet of Things," *Secur. Commun. Netw.*, vol. 2019, Jul. 2019, Art. no. 4301409. [Online]. Available: https://www.scopus.com

[5] A. Bilen and A. B. Özer, "Cyber-attack method and perpetrator prediction using machine learning algorithms," *PeerJ Comput. Sci.*, vol. 7, p. e475, Apr. 2021, doi: 10.7717/PEERJ-CS.475.

[6] J. Cho, "Efficient autonomous defense system using machine learning on edge device," *Comput., Mater. Continua*, vol. 70, no. 2, pp. 3565–3588, 2022, doi: 10.32604/cmc.2022.020826.

[7] C. M. Sayan, "An intelligent security assistant for cyber security operations," in *Proc. IEEE 2nd Int. Workshops Found. Appl. Self* Syst. (FAS*W)*, Sep. 2017, pp. 375–376, doi: 10.1109/FAS-W.2017.179.

[8] A. S. Gowri and P. S. i. Bala, "An agent based resource provision for IoT through machine learning in fog computing," in *Proc. IEEE Int. Conf. Syst., Comput., Autom. Netw. (ICSCAN)*, Mar. 2019, pp. 1–5, doi: 10.1109/ICSCAN.2019.8878821.

[9] S. Pundir, M. Wazid, D. P. Singh, A. K. Das, J. J. P. C. Rodrigues, and Y. Park, "Intrusion detection protocols in wireless sensor networks integrated to Internet of Things deployment: Survey and future challenges," *IEEE Access*, vol. 8, pp. 3343–3363, 2020, doi: 10.1109/ACCESS.2019.2962829.

[10] L. N. Tidjon, M. Frappier, and A. Mammar, "Intrusion detection systems: A cross-domain overview," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 4, pp. 3639–3681, 4th Quart., 2019, doi: 10.1109/COMST.2019.2922584.

[11] B. Sezari, D. P. F. Möller, and A. Deutschmann, "Anomaly-based network intrusion detection model using deep learning in airports," in *Proc. 17th IEEE Int. Conf. Trust, Secur. Privacy Comput. Communications/ 12th IEEE Int. Conf. Big Data Sci. Eng. (TrustCom/BigDataSE)*, Aug. 2018, pp. 1725–1729, doi: 10.1109/TrustCom/BigDataSE.2018.00261.

[12] M. A. Ferrag, L. Shu, O. Friha, and X. Yang, "Cyber security intrusion detection for agriculture 4.0: Machine learning-based solutions, datasets, and future directions," *IEEE/CAA J. Autom. Sinica*, vol. 9, no. 3, pp. 407–436, Mar. 2022, doi: 10.1109/JAS.2021.1004344.

[13] G. Liu, H. Zhao, F. Fan, G. Liu, Q. Xu, and S. Nazir, "An enhanced intrusion detection model based on improved KNN in WSNs," *Sensors*, vol. 22, no. 4, p. 1407, Feb. 2022, doi: 10.3390/s22041407.

[14] O. Kompougias, D. Papadopoulos, E. Mantas, A. Litke, N. Papadakis, D. Paraschos, A. Kourtis, and G. Xylouris, "IoT botnet detection on flow data using autoencoders," in *Proc. IEEE Int. Medit. Conf. Commun. Netw. (MeditCom)*, Sep. 2021, pp. 506–511, doi: 10.1109/MeditCom49071.2021.9647639.

[15] M. Husamuddin and M. Qayyum, "Internet of Things: A study on security and privacy threats," in *Proc. 2nd Int. Conf. Anti-Cyber Crimes (ICACC)*, Mar. 2017, pp. 93–97, doi: 10.1109/Anti-Cybercrime.2017.7905270.

[16] B. B. Zarpelão, R. S Miani, C. T. Kawakani, and S. C. de Alvarenga, "A survey of intrusion detection in Internet of Things," *J. Netw. Comput. Appl.*, vol. 84, pp. 25–37, Apr. 2017, doi: 10.1016/j.jnca.2017.02.009.

[17] C. Anita S. and S. Gupta, "An effective model for anomaly IDS to improve the efficiency," in *Proc. Int. Conf. Green Comput. Internet Things (ICGCIoT)*, Oct. 2015, pp. 190–194, doi: 10.1109/ICGCIoT.2015.7380455.

[18] N. Koroniotis, N. Moustafa, E. Sitnikova, and B. Turnbull, "Towards the development of realistic botnet dataset in the Internet of Things for network forensic analytics: Bot-IoT dataset," *Future Gener. Comput. Syst.*, vol. 100, pp. 779–796, Nov. 2019, doi: 10.1016/j.future.2019.05.041. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167739X18327687

[19] K. S. Kiran, R. K. Devisetty, N. P. Kalyan, K. Mukundini, and R. Karthi, "Building a intrusion detection system for IoT environment using machine learning techniques," *Proc. Comput. Sci.*, vol. 171, pp. 2372–2379, Jan. 2020, doi: 10.1016/j.procs.2020.04.257. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1877050920312497

[20] O. Almomani, M. A. Almaiah, A. Alsaaidah, S. Smadi, A. H. Mohammad, and A. Althunibat, "Machine learning classifiers for network intrusion detection system: Comparative study," in *Proc. Int. Conf. Inf. Technol. (ICIT)*, Jul. 2021, pp. 440–445, doi: 10.1109/ICIT52682.2021.9491770.

[21] A. Sirisha, K. Chaitanya, K. V. S. S. R. Krishna, and S. S. Kanumalli, "Intrusion detection models using supervised and unsupervised algorithms—A comparative estimation," *Int. J. Saf. Secur. Eng.*, vol. 11, no. 1, pp. 51–58, Feb. 2021, doi: 10.18280/ijsse.110106.

[22] Z. K. Maseer, R. Yusof, N. Bahaman, S. A. Mostafa, and C. F. M. Foozy, "Benchmarking of machine learning for anomaly based intrusion detection systems in the CICIDS2017 dataset," *IEEE Access*, vol. 9, pp. 22351–22370, 2021, doi: 10.1109/ACCESS.2021.3056614.

[23] H. Ao, "Using machine learning models to detect different intrusion on NSL-KDD," in *Proc. IEEE Int. Conf. Comput. Sci., Artif. Intell. Electron. Eng. (CSAIEE)*, Aug. 2021, pp. 166–177, doi: 10.1109/CSAIEE54046.2021.9543241.

[24] M. Shafiq, Z. Tian, Y. Sun, X. Du, and M. Guizani, "Selection of effective machine learning algorithm and Bot-IoT attacks traffic identification for Internet of Things in smart city," *Future Gener. Comput. Syst.*, vol. 107, pp. 433–442, Jun. 2020, doi: /10.1016/j.future.2020.02.017. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167739X19334880

[25] F. Anwar and S. Saravanan, "Comparison of artificial intelligence algorithms for IoT botnet detection on apache spark platform," *Proc. Comput. Sci.*, vol. 215, pp. 499–508, Jan. 2022, doi: 10.1016/j.procs.2022.12.052. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1877050922021238

[26] H. Lawrence, U. Ezeobi, O. Tauil, J. Nosal, O. Redwood, Y. Zhuang, and G. Bloom, "CUPID: A labeled dataset with pentesting for evaluation of network intrusion detection," *J. Syst. Archit.*, vol. 129, Aug. 2022, Art. no. 102621, doi: 10.1016/j.sysarc.2022.102621. [Online]. Available: https ://www.sciencedirect.com/science/article/pii/S1383762122001515

[27] T. Gaber, A. El-Ghamry, and A. E. Hassanien, "Injection attack detection using machine learning for smart IoT applications," *Phys. Commun.*, vol. 52, p. 101685, Jun. 2022, doi: 10.1016/j.phycom.2022.101685. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1874490722000490

[28] G. Rohini, C. G. Kousalya, and J. Bino, "Intrusion detection system with an ensemble learning and feature selection framework for IoT networks," *IETE J. Res.*, pp. 1–17, Aug. 2022, doi: 10.1080/03772063.2022.2098187.

[29] M. S. Islam, I. Hussain, M. M. Rahman, S. J. Park, and M. A. Hossain, "Explainable artificial intelligence model for stroke prediction using EEG signal," *Sensors*, vol. 22, no. 24, p. 9859, Dec. 2022, doi: 10.3390/s22249859.

[30] I. Hussain and S. J. Park, "Big-ECG: Cardiographic predictive cyber-physical system for stroke management," *IEEE Access*, vol. 9, pp. 123146–123164, 2021, doi: 10.1109/ACCESS.2021.3109806.

[31] I. Hussain and S. J. Park, "HealthSOS: Real-time health monitoring system for stroke prognostics," *IEEE Access*, vol. 8, pp. 213574–213586, 2020, doi: 10.1109/ACCESS.2020.3040437.

[32] T. Janarthanan and S. Zargari, "Feature selection in UNSW-NB15 and KDDCUP'99 datasets," in *Proc. IEEE 26th Int. Symp. Ind. Electron. (ISIE)*, Jun. 2017, pp. 1881–1886, doi: 10.1109/ISIE.2017.8001537.

[33] C. Beazley, K. Gadiya, R. K. U. Rakesh, D. Roden, B. Ye, B. Abraham, D. E. Brown, and M. Veeraraghavan, "Exploratory data analysis of a unified host and network dataset," in *Proc. Syst. Inf. Eng. Design Symp. (SIEDS)*, Apr. 2019, pp. 1–5, doi: 10.1109/SIEDS.2019.8735640.

[34] D. K. Bhattacharyya and J. K. Kalita. (2013). *Network Anomaly Detection: A Machine Learning Perspective*. [Online]. Available: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85053983816 &partnerID=40&md5=d08c13eb685e592ea4d6bac426f6b1f0

[35] S. Q. A. Shah, F. Z. Khan, and M. Ahmad, "The impact and mitigation of ICMP based economic denial of sustainability attack in cloud computing environment using software defined network," *Comput. Netw.*, vol. 187, Mar. 2021, Art. no. 107825, doi: 10.1016/j.comnet.2021.107825. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S138912862100013X

[36] P. Kumari and A. K. Jain, "A comprehensive study of DDoS attacks over IoT network and their countermeasures," *Comput. Secur.*, vol. 127, Apr. 2023, Art. no. 103096, doi: 10.1016/j.cose.2023.103096. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167404823000068

[37] A. Gupta and L. S. Sharma, "Detecting attacks in high-speed networks: Issues and solutions," *Inf. Secur. J., A Global Perspective*, vol. 29, no. 2, pp. 51–61, Mar. 2020, doi: 10.1080/19393555.2020.1722296.

[38] S. Hijazi, M. S. Obaidat, and S. Obaidat, "Address resolution protocol spoofing attacks and security approaches: A survey," *Secur. Privacy*, vol. 6, no. 3, 2018, doi: 10.1002/spy2.49.

[39] M. A. A. Ghamdi, "An optimized and secure energy-efficient blockchain-based framework in IoT," *IEEE Access*, vol. 10, pp. 133682–133697, 2022, 10.1109/ACCESS.2022.3230985.

[40] S. B. Wankhede, "Study of network-based DoS attacks," in *Nanoelectronics, Circuits and Communication Systems*, V. Nath and J. K. Mandal, Eds. Singapore: Springer, 2019.

[41] S. Saito, K. Maruhashi, M. Takenaka, and S. Torii, "TOPASE: Detection and prevention of brute force attacks with disciplined IPs from IDS logs," *J. Inf. Process.*, vol. 24, no. 2, pp. 217–226, 2016, doi: 10.2197/ipsjjip.24.217.

[42] N. Naik and P. Jenkins, ‘‘Discovering hackers by stealth: Predicting fingerprinting attacks on honeypot systems,’’ in *Proc. IEEE Int. Syst. Eng. Symp. (ISSE)*, Oct. 2018, pp. 1–8, doi: 10.1109/SysEng.2018.8544408.

[43] A. Sahi, D. Lai, Y. Li, and M. Diykh, ‘‘An efficient DDoS TCP flood attack detection and prevention system in a cloud environment,’’ *IEEE Access*, vol. 5, pp. 6036–6048, 2017, doi: 10.1109/ACCESS.2017.2688460.

[44] R. Gangula, V. M. Mohan, and R. Kumar, ‘‘A comprehence study of DDoS attack detecting algorithm using GRU-BWFA classifier,’’ *Meas., Sensors*, vol. 24, Dec. 2022, Art. no. 100570, doi: 10.1016/j.measen.2022.100570. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2665917422002045

[45] M. Catillo, A. Pecchia, and U. Villano, ‘‘No more DoS? An empirical study on defense techniques for web server denial of service mitigation,’’ *J. Network Comput. Appl.*, vol. 202, 2022, Art. no. 103363, doi: /10.1016/j.jnca.2022.103363. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1084804522000303

[46] S. A. Abdullah, ‘‘SEUI-64, bits an IPv6 addressing strategy to mitigate reconnaissance attacks,’’ *Eng. Sci. Technol., Int. J.*, vol. 22, no. 2, pp. 667–672, Apr. 2019, doi: 10.1016/j.jestch.2018.11.012. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2215098618312977

[47] W. Chang, Y. Liu, Y. Xiao, X. Xu, S. Zhou, X. Lu, and Y. Cheng, ‘‘Probability analysis of hypertension-related symptoms based on XGBoost and clustering algorithm,’’ *Appl. Sci.*, vol. 9, no. 6, p. 1215, 2019, doi: 10.3390/app9061215. [Online]. Available: https://www.mdpi.com/2076-3417/9/6/1215

[48] A. Géron, *Hands-On Machine Learning With Scikit-Learn, Keras, and TensorFlow*, 2nd ed. Sebastopol, CA, USA: O'Reilly Media, 2019.

[49] A. Abbas, M. A. Khan, S. Latif, M. Ajaz, A. A. Shah, and J. Ahmad, ‘‘A new ensemble-based intrusion detection system for Internet of Things,’’ *Arabian J. Sci. Eng.*, vol. 47, no. 2, pp. 1805–1819, Feb. 2022, doi: 10.1007/s13369-021-06086-5.

[50] L. Liu, P. Wang, J. Lin, and L. Liu, ‘‘Intrusion detection of imbalanced network traffic based on machine learning and deep learning,’’ *IEEE Access*, vol. 9, pp. 7550–7563, 2021, doi: 10.1109/ACCESS.2020.3048198.

[51] F. Rustam, M. Khalid, W. Aslam, V. Rupapara, A. Mehmood, and G. S. Choi, ‘‘A performance comparison of supervised machine learning models for covid-19 tweets sentiment analysis,’’ *PLoS One*, vol. 16, no. 2, Feb. 2021, Art. no. e0245909, doi: 10.1371/journal.pone.0245909.

[52] A. Thakkar and R. Lohiya, ‘‘Attack classification using feature selection techniques: A comparative study,’’ *J. Ambient Intell. Humanized Comput.*, vol. 12, no. 1, pp. 1249–1266, Jan. 2021, doi: 10.1007/S12652-020-02167-9. [Online]. Available: https://bbibliograficas.ucc.edu.co:2201/article/10.1007/s12652-020-02167-9

[53] M. C. Belavagi and B. Muniyal, ‘‘Multi class machine learning algorithms for intrusion detection—A performance study,’’ *Commun. Comput. Inf. Sci.*, vol. 746, pp. 170–178, Nov. 2017, doi: 10.1007/978-981-10-6898-0_14.

[54] S. Bose, A. Pal, R. SahaRay, and J. Nayak, ‘‘Generalized quadratic discriminant analysis,’’ *Pattern Recognit.*, vol. 48, no. 8, pp. 2676–2684, Aug. 2015, doi: 10.1016/j.patcog.2015.02.016. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S003132031500076X

[55] J. Ma, Y. Ding, J. C. P. Cheng, Y. Tan, V. J. L. Gan, and J. Zhang, ‘‘Analyzing the leading causes of traffic fatalities using XGBoost and grid-based analysis: A city management perspective,’’ *IEEE Access*, vol. 7, pp. 148059–148072, 2019, doi: 10.1109/ACCESS.2019.2946401.

[56] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, A. Müller, J. Nothman, G. Louppe, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, ‘‘Scikit-learn: Machine learning in Python,’’ *J. Mach. Learn. Res.*, vol. 12, pp. 2825–2830, Oct. 2012.

[57] L. Buitinck, G. Louppe, M. Blondel, F. Pedregosa, A. Mueller, O. Grisel, V. Niculae, P. Prettenhofer, A. Gramfort, J. Grobler, R. Layton, J. VanderPlas, A. Joly, B. Holt, and G. Varoquaux, ‘‘API design for machine learning software: Experiences from the scikit-learn project,’’ in *Proc. ECML PKDD Workshop, Lang. Data Mining Mach. Learn.*, 2013, pp. 108–122.

[58] H. M and S. M. N, ‘‘A review on evaluation metrics for data classification evaluations,’’ *Int. J. Data Mining Knowl. Manage. Process*, vol. 5, no. 2, pp. 01–11, Mar. 2015, doi: 10.5121/ijdkp.2015.5201.

[59] H. B Arteaga-Arteaga, A. Mora-Rubio, F. Florez, N. Murcia-Orjuela, C. E. Diaz-Ortega, S. Orozco-Arias, M. Delapava, M. A. Bravo-Ortíz, M. Robinson, P. Guillen-Rondon, and R. Tabares-Soto, ‘‘Machine learning applications to predict two-phase flow patterns,’’ *PeerJ Comput. Sci.*, vol. 7, p. e798, Nov. 2021.

[60] D. M. W. Powers, ‘‘Evaluation: From precision, recall andF-measure to ROC, informedness, markedness and correlation,’’ 2020, *arXiv:2010.16061*.

[61] B. Bengfort and R. Bilbro, ‘‘Yellowbrick: Visualizing the scikit-learn model selection process,’’ *J. Open Source Softw.*, vol. 4, no. 35, p. 1075, Mar. 2019, doi: 10.21105/joss.01075. [Online]. Available: https://joss.theoj.org/papers/10.21105/joss.01075

[62] S. M. Lundberg and S.-I. Lee, ‘‘A unified approach to interpreting model predictions,’’ in *Advances in Neural Information Processing Systems*, vol. 30, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds. Red Hook, NY, USA: Curran Associates, 2017. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2017/file/8a20a8621978632d76c43dfd28b67767-Paper.pdf

[63] M. Verkerken, L. D'hooge, T. Wauters, B. Volckaert, and F. De Turck, ‘‘Towards model generalization for intrusion detection: Unsupervised machine learning techniques,’’ *J. Netw. Syst. Manage.*, vol. 30, no. 1, Jan. 2022, doi: 10.1007/s10922-021-09615-7.

GUTIERREZ-PORTELA FERNANDO received the degree in systems engineering from Universidad Antonio Nariño, Colombia, in 1998, with a specialization in teleinformatics with Universidad de Ibagué, in 2001, and the magister degree in open software from Universidad Autónoma de Bucaramanga and Universidad de Oberta de Catalunya, Colombia, in 2012. He is currently pursuing the Ph.D. degree in engineering with Universidad Autónoma de Bucaramanga. He is an Assistant Professor with the Systems Engineering Program, Universidad Cooperativa de Colombia, Colombia. His research interests include anomaly detection systems, machine learning, and cybersecurity.

ARTEAGA-ARTEAGA HAROLD BRAYAN received the B.S. degree in electronic engineering from Universidad Autónoma de Manizales, Colombia, where he is currently pursuing the Ph.D. degree in engineering. Since 2018, he has been actively engaged with the University's Research Group on Bioinformatics and Artificial Intelligence. His research interests include data science, machine learning, deep learning, bioinformatics, and automation.

ALMENARES MENDOZA FLORINA (Member, IEEE) received the M.Sc. degree in telematics and the Ph.D. degree from the University Carlos III of Madrid (UC3M), in 2003 and 2006, respectively. Since 2008, she has been an Associate Professor with the Department of Telematics Engineering, UC3M. Her research interests include trust and reputation management models, identity management, secure architectures and risk assessment, PQC, and cybersecurity. This research has been applied to ubiquitous computing and IoT, smart grids, smart cities, or cloud computing.

CALDERÓN-BENAVIDES LILIANA received the B.S. degree in systems engineering from Universidad Autónoma de Bucaramanga (UNAB), in 2001, the M.S. degree in computer science from the Instituto Tecnológico y de Estudios Superiores de Monterrey (ITESM), UNAB, and the Ph.D. degree in computer science and digital communication from Universitat Pompeu Fabra, Spain, in 2010. She is currently an Associate Professor with the Faculty of Systems Engineering, UNAB. She is also the Director of the Information Technology Research Group and a Researcher with the Center for Research in Engineering and Organizations, UNAB. She is the Institutional Strategy in Data Science Leader at UNAB.

ACOSTA-MESA HÉCTOR-GABRIEL received the B.Sc. degree in computer systems engineering from the Veracruz Institute of Technology, Mexico, in 1991, the M.Sc. degree in artificial intelligence from the Universidad Veracruzana, Xalapa, in 1997, and the Ph.D. degree in artificial intelligence from the Artificial Intelligence Vision Research Unit, Department of Psychology, University of Sheffield, U.K., in 2003. He is currently a Researcher with the Artificial Intelligence Research Institute, Universidad Veracruzana, and the National Council of Science and Technology (CONACyT) of Mexico (Level 1). He is a member of various review committees for journals and specialized projects. He is the author of various research articles specialized in medical image analysis using computer vision. His research interests include machine learning techniques and computer vision applications. He is the Scientific Editor of the magazine *Komputer Sapiens*.

TABARES-SOTO REINEL received the B.S. degree in electronic engineering from Universidad Nacional de Colombia, in 2009, the B.S. degree in systems and computer engineering from Universidad de Caldas, Colombia, in 2016, the M.S. degree in engineering from Universidad Nacional de Colombia, in 2017, and the Ph.D. degree in computer science from Universidad Autónoma de Manizales, Colombia. Since 2014, he has been a Coordinator with the Department of Electronics, Universidad Autónoma de Manizales. His main research interests include steganalysis, machine learning, deep learning, bioinformatics, and high-performance computing.