IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received June 24, 2021, accepted August 5, 2021, date of publication August 11, 2021, date of current version August 19, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3104113

An Advanced Intrusion Detection System for IIoT Based on GA and Tree Based Algorithms

SYDNEY MAMBWE KASONGO
Department of Industrial Engineering, Stellenbosch University, Stellenbosch 7600, South Africa
School for Data Science and Computational Thinking, Stellenbosch University, Stellenbosch 7599, South Africa
e-mail: sydneyk@sun.ac.za

ABSTRACT The evolution of the Internet and cloud-based technologies have empowered several organizations with the capacity to implement large-scale Internet of Things (IoT)-based ecosystems, such as Industrial IoT (IIoT). The IoT and, by virtue, the IIoT, are vulnerable to new types of threats and intrusions because of the nature of their networks. So it is crucial to develop Intrusion Detection Systems (IDSs) that can provide the security, privacy, and integrity of IIoT networks. In this research, we propose an IDS for IIoT that was implemented using the Genetic Algorithm (GA) for feature selection, and the Random Forest (RF) model was employed in the GA fitness function. The models used for the intrusion detection processes include classifiers such as the RF, Linear Regression (LR), Naïve Bayes (NB), Decision Tree (DT), Extra-Trees (ET), and Extreme Gradient Boosting (XGB). The GA-RF generated 10 feature vectors for the binary classification scheme and 7 feature vectors for the multiclass classification procedure. The UNSW-NB15 is used to assess the effectiveness and the robustness of our proposed approach. The experimental outcomes demonstrated that for the binary modeling process, the GA-RF achieved a test accuracy (TAC) of 87.61% and an Area Under the Curve (AUC) of 0.98, using a feature vector that contained 16 features. These results were superior to existing IDS frameworks.

INDEX TERMS Internet of Things, intrusion detection, genetic algorithm, machine learning.

I. INTRODUCTION

In recent years, the Internet of Things (IoT) paradigm has shown massive adoption by different industries including the medical sector, vehicle manufacturers, home appliances manufacturers, etc. The acceptance of IoT technology has significantly changed the way we live [1]. The specific use of IoT in the modern industry gave birth to the Industrial IoT (IIoT) concept. Modern Industrial Internet of Things (I-IoT or IIoT) depicts using the regular IoT in different industrial ventures and organizations. IIoT contains countless actuators, sensors, control systems, communication and integration interfaces, advanced security systems, vehicular networks, home appliances networks, etc. All the nodes within the IIoT can connect to the Internet. Using IIoT in modern industries has greatly enhanced the capabilities of various sectors such as manufacturing plants, asset management systems, advanced logistics systems, etc. Moreover, the IIoT allows for several applications, devices, and services to connect the physical space to a virtual one [2].

There exist several ways IIoT nodes connect to the Internet and this includes communication protocols such as the Transmission Control Protocol and the Internet Protocol (TCP/IP) using Message Queue Telemetry Transport (MQTT), Modbus TCP, Cellular, Long-Range Radio Wide Area Network (LoRaWAN), etc. [3], [4]. Moreover, most IIoT nodes can collect, process, and transmit data. These abilities make them susceptible to some privacy and security threats that have the potential to jeopardize the IIoT systems and the applications to which they belong [5]. One of the key attributes of IIoT nodes is that they are always active while performing the collection, processing, and transmission of data. Fig. 1 depicts all the layers that are present in the IIoT, namely, the perceptual layer, the network layer, the application layer, and the Cloud. These layers are based on the flow of data. Moreover, each layer is prone to various types of attacks and intrusions that could compromise the systems within the IIoT. Some common attacks and intrusions on the IIoT ecosystem include access control

The associate editor coordinating the review of this manuscript and approving it for publication was Eyhab Al-Masri.

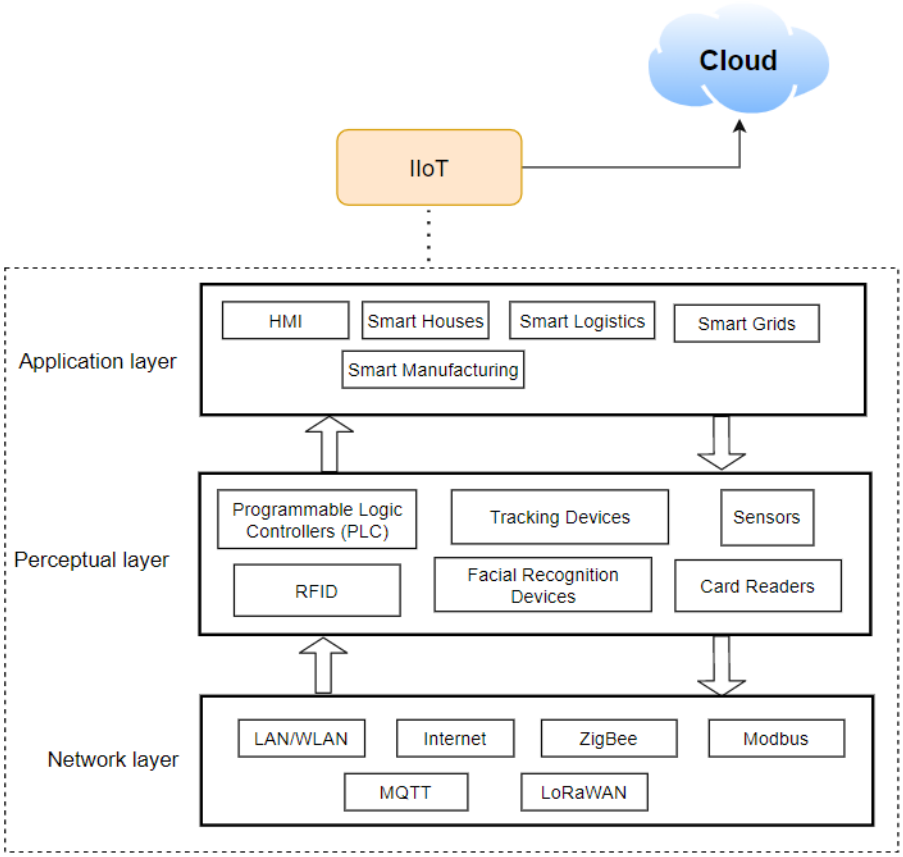

FIGURE 1. Typical IIoT architecture.

attacks, data corruption breaches, spoofing attacks, Denial of Service (DoS) attacks, Distributed DoS, Operating System (OS) attacks, jamming attacks, etc. To counter these malicious attacks and to guarantee that the active nature of IIoT nodes and the security of IIoT networks are maintained, a lot of organizations are implementing Intrusion Detection Systems (IDSs). Moreover, these IDSs can be configured at any layer in Fig. 1 [5].

An IDS plays a critical role in the IIoT by guaranteeing that the integrity, security, and privacy of data transmitted through its network are maintained. An IDS can prevent, detect, react and report any attacks or malicious activities that have the potential to cripple an IIoT network [6]. Traditional IDSs are broadly categorized as follows: signature-based, anomaly-based, and hybrid-based. Signature-based IDSs are designed using existing (known) attack signatures that can be found in the IDS database. Anomaly-based IDS are implemented using abnormal patterns within a network. Hybrid-based IDSs combine signature and anomaly-based IDSs. Some drawbacks of traditional IDSs include a high false-positive rate and a low detection accuracy. Additionally, they cannot detect novel types of intrusions and are incapable of preventing events such as zero-day attacks. To improve on the performance of traditional IDSs, researchers have explored the use of Artificial Intelligence (AI) and more particularly, the application of Machine Learning (ML) based techniques for IDS [7], [8].

ML is a branch of Artificial Intelligence (AI) that empowers various systems with the ability and the capacity to learn from experience and to ameliorate their decision-making process without any explicit programming [9]. At the top level, ML approaches are categorized as supervised and unsupervised. At a granular level, ML algorithms are classified as follows: supervised, unsupervised, semi-supervised, and reinforcement. Supervised ML methods improve their decision-making process by learning from a labeled dataset (a dataset with data points that have a label) to perform future predictions. In contrast, unsupervised ML approaches are used when the learning task involves unlabelled data. Semi-supervised ML algorithms use both labeled and unlabeled data during the learning process. Reinforcement ML methods compute rewards or errors based on their interaction within a given environment [10].

In this research, we propose an IDS for IIoT that uses Tree-based supervised ML algorithms. ML-based IDSs are generally trained using the latest intrusion detection datasets. Nonetheless, the majority of the modern datasets are large, both on the feature space dimension as well as the number of network traces. A high number of features in a dataset has the potential to negatively impact the training process of

ML algorithms. Often the performance of ML methods is reduced as the number of features increases. In other words, it is harder to perform the learning process as the number of attributes increases in a dataset [11]. Thus, it is crucial to perform a feature selection or extraction process to guarantee that the size of the attribute vector is reduced to an optimal number of required features [12].

There are three types of feature selection (FS) methods: wrapper-based FS, filter-based FS, and hybrid-based FS. In the instance of the filter-based FS method, the selection process relies on the nature of the data and it uses a variety of statistical methods to extract the optimal feature vector. The filter-based FS method is computationally cheap and efficient. In contrast, the wrapper-based FS approach employs a predictor in the selection process. This occurs by iteratively computing the predictor's performance over several subsets of features until the candidate optimal feature vector is found. The wrapper-based FS method is computationally expensive, but it is precise in comparison to other FS methods. The hybrid-based FS technique, sometimes called embedded-based FS, combines the filter-based and the wrapper-based FS methods [13]–[15]. In this research, we propose a wrapper-based FS method, based on the Genetic Algorithm (GA) [16] that uses the Random Forest (RF) ML algorithm [17] in its fitness function to generate optimal candidate feature vectors. Furthermore, to assess the performance of our proposed method, we use the UNSW-NB15 intrusion detection dataset. This dataset is widely adopted by the research community [18], [19]. The network traces present in the dataset were generated in a laboratory environment. But, they do mimic the real-world network traffic patterns, such as the ones generated by an IIoT network system [20]. Additionally, the UNSW-NB15 is a more complex dataset in comparison to the NSL-KDD or KDD Cup 99 datasets [20] and it includes a higher variety of network traffic patterns. Moreover, the UNSW-NB15 is a general-purpose dataset that paved the way to datasets such as the TON_IoT dataset [21].

The major goals and contributions of this paper are as follows:

- Firstly, we propose a Genetic Algorithm (GA)-based feature selection algorithm. The fitness function used in the GA method used the Random Forest (RF) to generate the fitness scores.
- Secondly, for each solution (attribute vector), we implement Tree-based algorithms such as RF, the Decision Tree (DT), and the Extra Tree (ET) methods. Moreover, the generated attribute vectors can be applied by other researchers using their own classifiers.
- Lastly, we conduct a comparison between our proposed method with existing systems. The results demonstrate a noteworthy improvement in performance.

The remainder of the paper is structured as follows. Section II presents an account of related work. Section III introduces the UNSW-NB15 dataset. Section IV presents the proposed IDS methodology. Section V outlines the experiments and provides discussions about the results. Section VI concludes this paper and provides future directions.

II. RELATED WORK

This section provides an account of related research works that were conducted in the domain of IDS using ML techniques. Moreover, this section serves as a survey of various IDS frameworks and solutions that were previously implemented for intrusion detection in IoT-based systems.

Liu *et al.* [22] implemented an IDS system for IoT using a Particle Swarm Optimization (PSO)-based technique for feature selection and the Support Vector Machine(SVM) ML algorithm for classification. The PSO method used in this research is based on the Light Gradient Boosting Machine (LightGBM). The authors used the UNSW-NB15 dataset to validate their model and they considered the accuracy and the False Alarm Rate (FAR) as the performance metrics. The experimental results demonstrated that the PSO-LightGBM achieved an overall accuracy of 86.68% and a high FAR of 10.62%. This research was based on the binary classification scheme. But, the authors could have also implemented the multiclass classification procedure to assess the full potential of their method. Moreover, the FAR obtained by the LightGBM is high.

Zhou *et al.* [23] implemented a Variational LSTM (VLSTM) IDS for Industrial Big Data systems. The VLSTM was implemented in conjunction with a feature selection and retention technique based on the reconstructed rendering of features. The authors used an Auto-Encoder Neural Network (AENN) to retrieve the low-dimensional attribute characteristics from high-dimensional datasets. To study their model, the researchers used the UNSW-NB15 dataset. During the evaluation phase, the following performance metrics were employed: the False Alarm Rate (FAR), the Area Under the Curve (AUC), the precision, the recall, and the F1-Score. The experimental results demonstrated that the VLSTM achieved an AUC of 0.895, a precision of 86%, a recall of 97.8%, and an F1-Score of 90.7%. Although these results were superior to some of the existing methods. The authors conceded that further experiments needed to be done to deal with the highly imbalanced nature of the UNSW-NB15.

In [24], the authors proposed an ML-based IDS using an adaptive principal component (APAC) for the feature selection process and an incremental extreme learning machine (IELM) algorithm for classification. In this research, the APAC is used to adaptively generate candidate attributes that are then fed to the IELM for the classification procedure. The authors considered the NSL-KDD and the UNSW-NB15 datasets to gauge the effectiveness of the presented framework. Moreover, the multiclass classification scheme was used for both datasets. The main performance metric that was utilized in this work was the accuracy achieved by a model on test data. In the case of the NSL-KDD dataset, the APAC-IELM achieved an accuracy of 81.22%. For the UNSW-NB15, the APAC-IELM obtained an accuracy

of 70.51%. Although the authors claimed that the obtained results were superior to those obtained by the existing systems, they conceded that more research needed to be undertaken to adapt the APAC-IELM to industrial control systems (ICS).

In [25], the authors proposed a deep neural network (DNN)-based IDS. In this research, the aim was to develop a flexible and robust IDS that could easily detect novel forms of attacks. To assess the efficacy of the presented method, the following datasets were considered: KDD-Cup99, UNSW-NB15, NSL-KDD, Kyoto, WSN-DS, and CICIDS 2017. The experimental processes were executed over 1000 epochs for each dataset. Focusing on the UNSW-NB15, the experiments demonstrated that the DNN obtained an accuracy of 76.1%, a precision of 95.1%, a recall of 96.3%, and F1-Score of 79.7% for the binary modeling process. In contrast, the DNN obtained an accuracy of 65.1%, an F1-Score of 75.6%, a precision of 59.7%, and a recall of 65.1% for the multiclass modeling procedure.

Hanif *et al.* [26] presented an IDS for IoT networks using artificial neural networks (ANN). This system was implemented to overcome the issue of security that is a major concern in IoT networks. Given the fact that IoT devices often lack the capacity to perform high-level computation for security, the authors decided to explore the possibility of using an ML-based IDS system as the first line of defense. To assess the effectiveness of the proposed method, the authors utilized the UNSW-NB15. The experimental outcomes claimed that the ANN-IDS obtained a precision score of 84.00% for the binary classification process. However, the researchers did not provide much clarity on how the hyper-parameters of the ANN were tuned to arrive at their conclusion. Moreover, the authors did not consider any feature selection method.

In [20], the authors conducted a complexity comparison analysis between the UNSW-NB15 and the KDD99 datasets. To achieve the comparison, the authors used various methods, including the expectation-maximization (EM) clustering algorithm and the ANN methods. In this work, the models were assessed using the FAR and the accuracy. In the instance of the KDD99, the EM clustering achieved an accuracy of 78.06% and a FAR of 23.79%. In contrast for the UNSW-NB15, the EM clustering obtained a FAR of 23.79% and an accuracy of 78.47%. Furthermore, the ANN technique attained an accuracy of 81.34% and a FAR of 21.13% when tested on the UNSW-NB15. This research concluded that the UNSW-NB15 dataset is more complex in contrast to the KDD99 dataset.

Ketzaki [27] proposed a light-weight IDS using ANN. This system is destined to secure modern communication systems (5G networks, IIoT networks, etc.). The ANN-IDS presented in this research was designed in two stages. The first stage is the feature extraction procedure using statistical analysis. The second step is the classification process. The authors considered the binary classification scheme using the UNSW-NB15 intrusion detection dataset. The performance metric used to evaluate the ANN models is the accuracy that was obtained on the test data. The results demonstrated that the best model attained an accuracy score of 83.9%. In their future endeavor, the authors aimed to improve the effectiveness of the proposed method.

In [28], the author presented an IDS framework using the J48 tree-based classifier and the SVM algorithm. Several methods were used to conduct the feature selection process, including the GA, the firefly optimization (FFA), and the grey wolf optimizer (GWO). The researchers used the UNSW-NB15 dataset to gauge the effectiveness of the models implemented in the experiments. The results showed that the accuracy scores obtained by the GA-J48, GWO-J48, and the FFA-J48 are 86.874%, 85.676%, and 86.037%, respectively. Moreover, the accuracy scores achieved by the GA-SVM, GWO-SVM, and FFA-SVM are 86.387%, 84.485%, and 85.429%, respectively. Although these are impressive results using the J48 and the SVM methods, the authors recommended that future work be conducted using other approaches such as deep learning methods.

In [29], the researchers implemented a novel feature selection method named Tabu Search - Random Forest (TS-RF). TS-RF is a wrapper-based feature extraction technique in which the TS algorithm conducts the attributes search and the RF approach is used as the learning method. To verify the performance of their model, the authors considered the UNSW-NB15 dataset. The main performance metrics were the accuracy and the False Positive Rate (FPR). The results demonstrated that the TS-RF in conjunction with the RF classifier obtained an accuracy of 83.12% and an FPR of 3.7%. Although the obtained results are promising, the authors conceded that they did not consider the class imbalance problem found in the UNSW-NB15 dataset.

In [30], a Two-Stage (TS) model for IDS was proposed. This methodology used the first stage to detect minority classes of intrusions and the second step to detect majority classes of attacks. The ML classification method used in this work is the RF method. The authors used the Information Gain (IG) for feature extraction. The IG-TS IDS was evaluated using the UNSW-NB15 dataset. The performance metrics considered in this research are accuracy and FAR. In their experiments, the authors used the binary classification scheme as their main configuration. The experimental results showed that the IG-TS obtained a FAR of 15.64 % and an accuracy of 85.78 %. In future works, the authors aimed to change the classifier that was utilized in the two stages.

In [31], the authors proposed an ML-based IDS using the GA algorithm and the Logistic Regression (LR) method for attributes selection. The binary classification process was conducted using a Tree-based classifier, namely the C4.5 method. The UNSW-NB15 was used to assess the efficacy of the presented method. The authors considered a number of performance metrics to evaluate the proposed approach, however, the accuracy that was obtained on test data was the main metric. The experimental results

showed that the GA-LR-DT attained an accuracy of 81.42%. This research did not demonstrate the effectiveness of the GA-LR-DT for the multiclass classification scheme.

Kasongo and Sun [32] proposed an IDS using an XGBoost (extreme gradient boosting) based feature extraction method in conjunction with several ML methods. The XGBoost, which is an ensemble-tree based algorithm, is used in this research to decrease the number of attributes in the UNSW-NB15. One of the classifiers used in this work is the LR method. The experimental results demonstrated that the XGBoost-LR achieved an accuracy of 75.51% and 72.53% for the binary and multiclass classification schemes, respectively. To overcome the class imbalance problems in the UNSW-NB15 dataset, the authors suggested using oversampling techniques.

In [33], the authors implemented an SVM-based NIDS using the UNSW-NB15 dataset. This system was designed to accommodate the unique nature of IoT networks. The authors considered the accuracy, the detection rate, and the false positive rate as the main performance metrics. The experiments were conducted for both the binary and multiclass classification schemes. The result showed that the SVM-NIDS attained an AC of 85.99% for the binary modeling task. In the instance of the multiple classes setting, the SVM-NIDS obtained an accuracy of 75.77%.

Kumar *et al.* [34] applied the UNSW-NB15 as an offline data source to design an ML-based IDS that would also be used to perform online intrusion detection. The authors used the Information Gain (IG) methodology for the feature selection procedure. The IG method selected 13 attributes. For the classification process, the researchers used an integrated approach that included the following Tree-based classifiers: C5, CHAID, CART, and QUEST. The outcome of the experiments demonstrated that the proposed system obtained an accuracy of 84.83% for the binary classification procedure. However, one of the drawbacks of the IDS presented here is its inability to detect unknown attacks. Solving this issue was one of the recommendations made by the authors.

In [35], the researchers presented an IDS using deep learning methods such as the Long-Short Term Memory (LTSM) RNN. To assess the effectiveness of the proposed approach, the authors used the UNSW-NB15 dataset. Moreover, the authors used the accuracy that was obtained during the classification task as the main performance metric. The experimental processes showed that the LSTM method obtained an accuracy of 85.42% for the binary modeling process. Although the authors claimed that these results were superior to existing ones, they did not consider implementing a feature selection algorithm.

Elijah *et al.* [36], proposed an ensemble and deep learning-based method for network intrusion detection. The LSTM algorithm was used to implement the deep learning model. The optimization algorithm applied to the LSTM is Stochastic Gradient Descent (SGD). The activation function applied in the LSTM layers is the Rectified Linear Unit (ReLU) in the instance of the binary classification task. For the multiclass classification scheme, the authors used the Softmax function. The UNSW-NB15 dataset was used in order to evaluate the performance of the proposed approach. The experimental results show that the LSTM IDS achieved an accuracy of 80.72% for the two-way classification procedure. In contrast, the LSTM IDS obtained an accuracy of 72.26% for the multiclass classification tasks.

In [37], the authors proposed a deep learning-based IDS using deep neural networks. This model was built using a combination of residual blocks (ResBlk). The ResBlks contain convolutional neural networks (CNNs) and recurrent neural networks (RNN). Moreover, the authors utilized the NSL-KDD and the UNSW-NB15 dataset to assess the performance of the proposed approach. The accuracy was one of the main performance metrics that was used to evaluate the outcome of the experiments. The results showed that the DL method achieved an accuracy of 99.21% and 86.64% in the instance of NSL-KDD and UNSW-NB15 datasets, respectively. Although these results are promising, the authors conceded that more experiments need to be conducted to improve the current performance numbers.

Assiri [38] proposed a GA-RF-based method for anomaly classification. In this work, the authors used the GA for attributes and parameters selection and the RF method for classification. Moreover, the researchers considered the binary classification scheme. The UNSW-NB15 was one of the datasets used to assess the performance of their model. The accuracy, recall, and precision were the main performance metrics that were utilized to evaluate the GA-RF presented here. The experimental results demonstrated that the GA-RF achieved a classification accuracy of 86.70%, a recall of 87.00%, and a precision of 87%.

In [39], the authors implemented an advanced IDS. This system was designed using a multi-objective feature selection method based on a special variation of the GA in conjunction with the Logistic regression (LR) algorithm. The RF method was one of the ML methods that were used to assess the performance of the proposed methodology. The UNSW-NB15 was amongst the datasets that were employed to evaluate the models. The accuracy was the main performance metric that was considered to gauge the effectiveness of the GA-LR-RF. The experimental outcomes demonstrated that the GA-LR-RF achieved an accuracy of 64.23% for the multiclass classification task.

III. THE UNSW-NB15 DATASET

The UNSW-NB15 [19] is an advanced dataset used for IDS research and it is widely used in the literature. The raw packets (network traces) contained in the UNSW-NB15 dataset were generated by the IXIA PerfectStorm tool in a laboratory set-up of the Cyber Range Laboratory of the Australian Center for Cybersecurity (ACCS). The UNSW-NB15 contains 42 attributes listed in Table 1. As depicted in the list of attributes in Table 1; 3 features are categorical in nature and 39 attributes are numerical (*binary*, *float* and *integer*).

TABLE 1. UNSW-NB15 dataset attributes list.

No.	Feature	Category	No.	Feature	Category
f1	dur	float	f22	dtcpb	integer
f2	proto	nominal	f23	dwin	integer
f3	service	nominal	f24	tcprtt	float
f4	state	nominal	f25	synack	float
f5	spkts	integer	f26	ackdat	float
f6	dpkts	integer	f27	smean	integer
f7	sbytes	integer	f28	dmean	integer
f8	dbytes	integer	f29	trans_depth	integer
f9	rate	float	f30	response_body_len	integer
f10	sttl	integer	f31	ct_srv_src	integer
f11	dttl	integer	f32	ct_state_ttl	integer
f12	sload	float	f33	ct_dst_ltm	integer
f13	dload	float	f34	ct_src_dport_ltm	integer
f14	sloss	integer	f35	ct_dst_sport_ltm	integer
f15	dloss	integer	f36	ct_dst_src_ltm	integer
f16	sinpkt	float	f37	is_ftp_login	binary
f17	dinpkt	float	f38	ct_ftp_cmd	integer
f18	sjit	float	f39	ct_flw_http_mthd	integer
f19	djit	float	f40	ct_src_ltm	integer
f20	swin	integer	f41	ct_srv_dst	integer
f21	stcpb	integer	f42	is_sm_ips_ports	binary

The UNSW-NB15 is composed of two datasets that include the UNSW-NB15-train and the UNSW-NB15-test. In this paper, UNSW-NB15-train is further divided into two datasets. The first one is the UNSW-NB15-75 that makes up 75% of the full UNSW-NB15-train. The second one is the UNSW-NB15-25 that accounts for 25% of the UNSW-NB15-train subset. In this study, UNSW-NB15-75 is used during the training phase of the models and the UNSW-NB15-25 is used during the validation phase of the models. It is crucial to perform a validation process to guarantee that the results that were obtained during the training phase are optimal. Moreover, the validation results *must* be like those of the training procedure. The entire UNSW-NB15-test dataset is used during the testing phase of the models presented in this research.

The UNSW-NB15 intrusion detection dataset contains the following nine categories of attacks [20]:

Fuzzers, Analysis, Exploits, Worms, Shellcode, DoS, Generic, Reconnaissance, and Backdoor. The value distribution of the UNSW-NB15 (UNSW-NB15-100), the UNSW-NB15-75, the UNSW-NB15-25, and the UNSW-NB15-TEST datasets are shown in Table 2.

TABLE 2. UNSW-NB15 dataset values distribution.

Attack Category	UNSW-NB15-100	UNSW-NB15-75	UNSW-NB15-25	UNSW-NB15-TEST
Normal	56000	41911	14089	37000
Generic	40000	30081	9919	18871
Exploits	33393	25034	8359	11132
Fuzzers	18184	13608	4576	6062
DoS	12264	9237	3027	4089
Reconnaissance	10491	7875	2616	3496
Analysis	2000	1477	523	677
Backdoor	1746	1330	416	583
Shellcode	1133	854	279	378
Worms	130	99	31	44

IV. THE PROPOSED IIoT IDS METHODOLOGY

The architecture of the proposed framework is depicted in Fig. 2 whereby there are three main phases, namely, the pre-processing phase, the feature selection phase, and the modeling and evaluation phases. In the pre-processing phase, we load the datasets (training set, validation set, and testing sets). Each dataset is cleaned and normalized. In the feature selection phase, the cleaned training dataset is used to compute the candidates feature vectors using the GA method in conjunction with the RF algorithm. In the modeling and evaluation step, the models (RF, EtraTrees, DT, LR, XGB) are trained using the cleaned training dataset with a particular attribute vector generated by the previous phase. Once the models have been trained, they are evaluated using the cleaned validation set and they are tested using the cleaned testing set. The building blocks of the proposed framework are explained in more detail in the next subsections.

A. PRE-PROCESSING PHASE

The most important aspects of the pre-processing phase are the cleaning and data normalization steps. Data cleaning is crucial because it ensures that the quality of the data used to build the models has been improved. The steps taken to clean the data include: removing duplicates, replacing missing data, fixing structural errors, and removing unwanted (potentially noisy) observations. Once, the data have been cleaned, they require normalization. In this research, we apply the Min-Max scaling [40] and it is defined as follows:

$$x_{norm} = (p - q)\frac{x_n - min(x_n)}{max(x_n) - min(x_n)} \tag{1}$$

where x represent a given feature in the feature space, X.

This scaling process acts as a safeguarding process by squeezing the values of each feature within a certain range.

B. RANDOM FOREST

The building blocks of the Random Forest (RF) algorithm are Decision Trees (DTs). A DT is a supervised ML method that is applied in tasks such as regression and classification. In simple terms, a DT algorithm uses a tree-like structure to compute the predictions. Each DT contains three types of nodes: namely, the root node, the internal nodes, and the category nodes. For a given input vector, the DT computes its prediction from the root node, traversing many internal nodes, to the category nodes [41], [42].

In this research, we use an RF classifier in the fitness function of the GA algorithm described in the next section. The RF algorithm was devised by L. Breiman [43] and it is one of the most widely used ML algorithms today. The RF algorithm is an ensemble of Decision Trees (DTs) classifiers whereby each individual DT is built using an attribute vector that is randomly selected from the input vector. Finally, each DT casts a vote for the most popular label in the selected input attribute vector. The label (class) with the highest score wins the poll [44], [45]. The RF method can be formulated as follows:

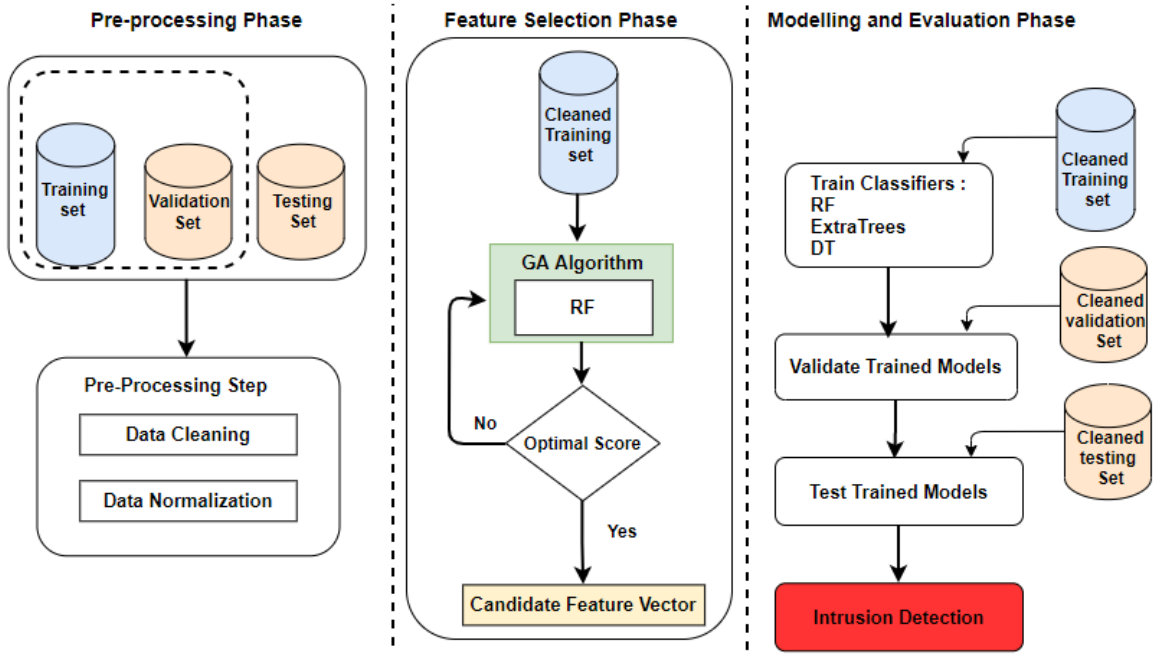

FIGURE 2. The proposed IDS framework for IIoT.

Let $P = \{X_1, y_1, \ldots, (X_k, y_k)\}$ be a training subset of inputs vectors and labels that are randomly selected given probability distribution (dataset), $(X_n, y_n) \sim (X, Y)$.

The aim is to compute a model (classifier) label y given an input X from P.

Let F, be a group of possibly weak classifiers defined as follows: $F = \{f_1(X), \ldots, f_N(X)\}$ where N is the total number of models. Each model, $f_n(X)$, in F is defined as a Decision Tree (DT). Therefore, F is called the Random Forest.

Each model $f_n(X)$ has some parameters defined as $B_n = (\beta_{n1}, \beta_{n2}, \ldots, \beta_{np})$. The notation of each tree in the forest becomes: $f_n(X) = f(X|B_n)$.

The attributes that appear in the nodes of the n^{th} DT are randomly selected based on B_n. The final result of the Forest, $f(X)$ (a combination of all the classifiers) is computed by majority voting. The label with the most votes is the output of the RF.

C. EXTRA-TREES

The Extra-Trees (ET) method is a tree-based algorithm (a meta-estimator) that is related to the RF algorithm because it also uses an ensemble of DTs to conduct the classification or the regression processes. However, unlike the RF algorithm, the ET approach randomly selects the nodes cut points. Therefore, the ET method adds another layer of randomization while maintaining its optimization capability [46].

D. FEATURE SELECTION PHASE USING GENETIC ALGORITHM

The Genetic Algorithm (GA) is an Evolutionary Algorithm (EA) that has gained popularity by solving various optimization problems with a low computational cost [47]. EAs are methods that are inspired by biological principles and are used for optimization or learning tasks. EAs have the following main traits [48]:

- **Population** EAs methods conserves a group of candidate solutions labelled *population*.
- **Fitness** An *individual* is a solution within a population. Each individual possesses its *code* (Gene representation) and its *fitness score*.
- **Variation** The individual goes through changes (mutations) similar to the biological genetic gene variation. This is how an EA algorithm performs the search in the solution space.

The main steps in the GA algorithm are as follows [49]:

1) Initialize the Population
2) Compute the fitness function
3) Perform the Selection
4) Perform the Crossover
5) Conduct the Mutation

In this research, the fitness function was implemented using the Random Forest algorithm presented in Algorithm 1.

Algorithm 2 depicts the steps (pseudo code) that were used to implement the GA algorithm on the UNSW-NB15 dataset. Moreover, Figure 3 simplifies this algorithm by outlining the major steps in a flowchart format.

E. MODELLING AND EVALUATION PHASE

1) PERFORMANCE METRICS

In this study, we used the following metrics to measure the performance of our proposed method: the accuracy (AC), the precision (PR), the recall (RC) and F1-Score (F1S) [50].

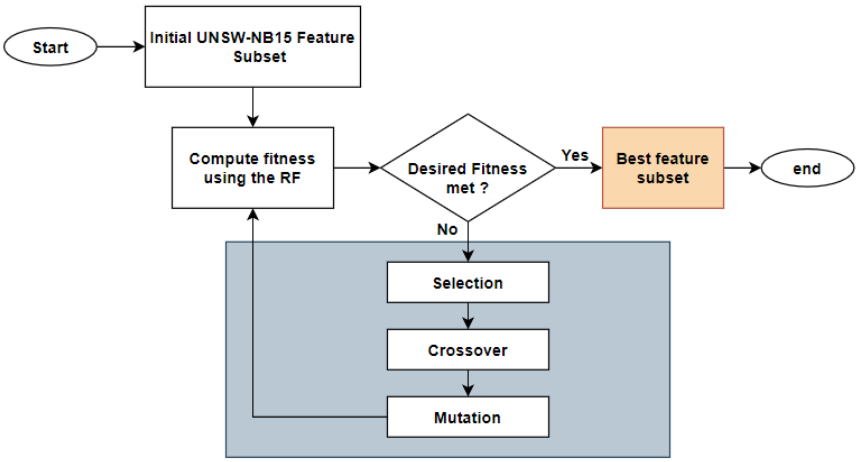

FIGURE 3. GA algorithm applied to the UNSW-NB15 dataset.

Algorithm 1 RF Algorithm in the GA Fitness Function

Input: X, y; the input dataframe and output series
Output: AC; the Accuracy obtained by the RF model
1. Spilt X and y in X_{train}, X_{val}, y_{train}, y_{val}
2. Instantiate rf, the model.
3. Fit rf using X_{train} and y_{train}
4. Evaluate rf using X_{val}
5. Compute predictions $y_{predictions}$
6. Compute AC using $y_{predictions}$ and y_{train}

Algorithm 2 GA Algorithm Applied on the UNSW-NB15

Require: D, the UNSW-NB15 data-frame
Require: F, an array that contains the feature names
Require: T, the target value
Require: L, an empty list to store the feature subset
Require: mi, maximum iteration
START
1. Initialize the population P, using F.
2. Implement the fitness function using RF
3. Compute the fitness using D, F, T and P
4. Compute optimal fitness value, v
5. Update L
for i in **range**(mi)
    6. Implement crossover
    7. Run mutations
    8. Compute the fitness
    9. Compute optimal fitness value, v
    10. Update L
end for
11. Convergence reached $\rightarrow L$ and v
STOP

The F1S represents the harmonic mean of the PR and RC. These metrics are chosen on the basis that we are faced with a classification problem. Moreover, in this research, we implement binary and multiclass classification processes. The AC, the RC, the PR, and the F1S are computed as follows:

$$AC = \frac{TP + TN}{TP + TN + TP + FN} \tag{2}$$

$$RC = \frac{TP}{TP + FN} \tag{3}$$

$$PR = \frac{TP}{TP + FP} \tag{4}$$

$$F1S = 2\frac{RC.PR}{RC + PR} \tag{5}$$

where each component in the above equations is defined as follows:

- True Positive (TP): represents the intrusions that are correctly labelled as attacks.
- True Negative (TN): normal network traces that are correctly labelled as legitimate.
- False Positive (FP): normal network traces that are labelled as intrusions.
- False Negative (FN): network intrusions that are wrongly labelled as non-intrusive (normal).

Additionally, to verify the efficacy of pour proposed method, we also plotted the receiver operating characteristic curve (ROC) curves for the models. The ROC curve plots the True Positive Rate (TPR) vs. the False Positive Rate (FPR) of a given model. The area under the ROC curve is defined as the Area Under the Curve (AUC). The value of the AUC is always between 0 and 1. An efficient model has an AUC value closer to 1 [51].

$$TPR = \frac{TP}{TP + FN} \tag{6}$$

$$FPR = \frac{FP}{FP + FN} \tag{7}$$

V. EXPERIMENTS AND DISCUSSIONS

A. EXPERIMENTAL CONFIGURATION

In this research, the experiments were conducted on a Laptop with the following specifications: DELL 153000 series

Windows 10 OS, Intel Core i7-8568U-CPU, 1.8GHz - 1.99 GHz. The ML framework that was used to implement the simulations is the Scikit-Learn (a Python-based framework) [52].

B. EXPERIMENTAL RESULTS AND DISCUSSIONS

In this research, the experiments were conducted in two phases (phase 1 and phase 2). In phase 1, we implemented the GA algorithm on the UNSW-NB15 dataset. This process generated two sets of feature vectors: V_b and V_m.

$$V_b = \{f_1, f_2, f_3, f_4, f_5, f_6, f_7, f_8, f_9, f_{10}\} \quad (8)$$

$$V_m = \{g_1, g_2, g_3, g_4, g_5, g_6, g_7\} \quad (9)$$

where V_b the group of possible solutions generated by the GA for the binary classification scheme and V_m denotes the group of possible solutions generated by the GA for the multiclass classification process. Table 3 and Table 4 provide the details about the vectors in V_b and V_m. These tables have three columns whereby the first one shows the vector name, the second column specifies the number of features that are present in the feature vector and the third column provides a list of features (attributes) that were selected by the GA.

In the second phase of our experiments, we implemented two classification processes. We first conducted the binary classification process whereby the target feature was binary (Normal or Attack). In this step, we considered all the feature vectors in V_b. We used the Logistic Regression (LR) [53] as our baseline model and we implemented the following Tree-based methods: DT, RF, ET, and XGB. The baseline model was used as our point of departure and the aim was to beat its performance using the other classifiers. The results of the experiments are presented in Table 5 – 14. The most optimal test accuracy (TAC), 87.61%, was achieved by the RF method using f_3, as shown in Table 7. Moreover, this model obtained a validation accuracy (VAC) of 95.87%, a recall (RC) of 98.34%, a precision (PR) of 82.51%, and an F1-score (F1S) of 89.73%. Moreover, for each of the classifiers that were evaluated using f_3, we computed the ROC curves. The results are depicted in Figure 3 whereby the RF achieved an AUC = 0.98. This value demonstrates that the quality of classification yielded by the RF is high. Although the TAC obtained by the XGB method (Table 7) was lower than that of the RF approach, it yielded an AUC = 0.98. This shows that the classification quality of the XGB classifier is high. Both the RF and the ET surpassed the AUC = 0.895 of the VLSTM presented in [23].

In the second step of phase 2, we implemented the multiclass classification process whereby all the labels (10 classes) present in the UNSW-NB15 were considered. Moreover, in this step, we utilized all the attribute vectors in V_m. The Naïve Bayes (NB) classifier [54] was used as the baseline model and we further implemented the following Tree-based algorithms: DT, RF, ET, and XGB. As mentioned in the previous step, the baseline model was utilized as our starting point and the goal was to surpass its performance using the other models. The outcomes are shown in Table 15 – 21. As depicted in Table 19, the experimental results demonstrated that the best model was the ET using g_5. It attained a VAC of 82.64%, a TAC of 77.64%, an RC of 83.09%, a PR of 77.64%, and F1S of 80.27%. Furthermore, we computed the confusion matrix to check how the model performed

TABLE 3. Features selected by the GA - Binary classification.

Feature vector	No. of features	list of features
f_1	21	dur, state, dpkts, sbytes, sload, dload, sloss, dloss, sjit, dwin, synack, smean, dmean, response_body_len, ct_srv_src, ct_dst_ltm, ct_src_dport_ltm, ct_dst_sport_ltm, ct_dst_src_ltm, ct_ftp_cmd, ct_srv_dst
f_2	17	service, sbytes, dbytes, sttl, sload, dload, sinpkt, dinpkt, swin, stcpb, synack, smean, trans_depth, ct_dst_ltm, ct_dst_src_ltm, ct_srv_dst, is_sm_ips_ports
f_3	16	dur, service, dpkts, sbytes, sttl, djit, smean, dmean, trans_depth, response_body_len, ct_src_dport_ltm, ct_dst_src_ltm, is_ftp_login, ct_ftp_cmd, ct_srv_dst, is_sm_ips_ports
f_4	13	dpkts, sbytes, sloss, dloss, sinpkt, djit, tcprtt, smean, dmean, ct_srv_src, ct_src_dport_ltm, ct_ftp_cmd, ct_srv_dst
f_5	18	dur, sbytes, sttl, dloss, sinpkt, djit, dtcpb, synack, ackdat, smean, dmean, ct_srv_src, ct_state_ttl, ct_src_dport_ltm, ct_dst_sport_ltm, ct_dst_src_ltm, is_ftp_login,ct_srv_dst
f_6	17	dur, service, sbytes, dbytes, sttl, sloss, dloss, sjit, stcpb, synack, ackdat, smean, dmean, ct_srv_src, ct_dst_src_ltm, ct_srv_dst, is_sm_ips_ports
f_7	20	service, spkts, sbytes, dttl, sload, dloss, sinpkt, djit, swin, stcpb, synack, ackdat, smean, dmean, ct_srv_src, ct_src_dport_ltm, ct_dst_sport_ltm, ct_dst_src_ltm, ct_flw_http_mthd, ct_srv_dst
f_8	27	dur, proto, service, dpkts, sbytes, dbytes, rate, dttl, sload, dload, dloss, sinpkt, sjit, djit, swin, stcpb, dwin, ackdat, smean, dmean, response_body_len, ct_srv_src, ct_dst_ltm, ct_src_dport_ltm, ct_dst_src_ltm, ct_flw_http_mthd, ct_srv_dst
f_9	16	dur, sbytes, dbytes, sttl, dttl, sjit, swin, dtcpb, tcprtt, smean, trans_depth, response_body_len, ct_srv_src, ct_dst_src_ltm, ct_ftp_cmd, is_sm_ips_ports
f_{10}	17	proto, dpkts, sbytes, dbytes, sttl, swin, tcprtt, synack, ackdat, smean, ct_dst_ltm, ct_src_dport_ltm, ct_dst_sport_ltm, ct_dst_src_ltm, is_ftp_login, ct_src_ltm, ct_srv_dst

TABLE 4. Features selected by the GA - Multiclass classification.

Feature vector	No. of features	list of features
g_1	22	service, spkts, sbytes, dbytes, rate, sttl, sloss, dinpkt, sjit, swin, tcprtt, synack, ackdat, smean, dmean, trans_depth, ct_state_ttl, ct_src_dport_ltm, is_ftp_login, ct_ftp_cmd, ct_src_ltm, ct_srv_dst
g_2	25	proto, service, state, dpkts, sbytes, dbytes, sttl, dttl, sloss, dloss, dinpkt, sjit, djit, stcpb, dwin, tcprtt, smean, dmean, trans_depth, ct_state_ttl, ct_dst_ltm, ct_ftp_cmd, ct_flw_http_mthd, ct_srv_dst, is_sm_ips_ports
g_3	28	dur, proto, service, state, spkts, dpkts, sbytes, dbytes, rate, sload, dload, sloss, sjit, swin, stcpb, dtcpb, dwin, tcprtt, ackdat, smean, dmean, ct_state_ttl, ct_src_dport_ltm, is_ftp_login, ct_ftp_cmd, ct_flw_http_mthd, ct_src_ltm, ct_srv_dst
g_4	20	proto, service, spkts, dpkts, sbytes, sload, dloss, sinpkt, dinpkt, sjit, djit, tcprtt, ackdat, smean, dmean, ct_srv_src, ct_state_ttl, ct_dst_sport_ltm, ct_dst_src_ltm, ct_flw_http_mthd
g_5	17	proto, service, spkts, dpkts, dbytes, sttl, dloss, dinpkt, sjit, tcprtt, smean, dmean, trans_depth, ct_dst_src_ltm, is_ftp_login, ct_ftp_cmd, is_sm_ips_ports
g_6	26	dur, proto, service, spkts, dpkts, sbytes, dbytes, sttl, dttl, dload, dloss, djit, stcpb, dtcpb, dwin, tcprtt, synack, ackdat, smean, dmean, ct_srv_src, ct_dst_ltm, ct_src_dport_ltm, ct_dst_sport_ltm, is_ftp_login, is_sm_ips_ports
g_7	18	proto, service, state, dpkts, sbytes, dbytes, sinpkt, swin, tcprtt, ackdat, smean, dmean, trans_depth, ct_state_ttl, ct_dst_src_ltm, is_ftp_login, ct_ftp_cmd, ct_flw_http_mthd

TABLE 5. Binary classification for f_1.

Model	FV	VAC	TAC	RC	PR	F1S
LR	f_1	88.63 %	73.40 %	91.76 %	69.60 %	79.16 %
DT	f_1	94.83 %	85.59 %	94.65 %	81.98 %	87.86 %
RF	f_1	95.95 %	86.89 %	98.50 %	81.53 %	89.22 %
ET	f_1	95.82 %	86.67 %	98.26 %	81.38 %	89.03 %
XGB	f_1	94.80 %	86.22 %	98.21 %	80.87 %	88.70 %

TABLE 6. Binary classification for f_2.

Model	FV	VAC	TAC	RC	PR	F1S
LR	f_2	87.90 %	74.09 %	90.56 %	70.65 %	79.38 %
DT	f_2	94.50 %	86.40 %	95.96 %	82.29 %	88.60 %
RF	f_2	95.86 %	87.37 %	98.69 %	82.02 %	89.59 %
ET	f_2	95.80 %	87.13 %	98.48 %	81.84 %	89.39 %
XGB	f_2	94.94 %	86.72 %	98.78 %	81.18 %	89.12 %

for each class present in the UNSW-NB15. As depicted in Figure 4, the ET performed optimally in detecting the following classes: Normal, Generic, Exploits, Dos, Reconnaissance, and Shellcode. However, the ET underperformed for some minority classes such as Worms, Backdoor, and Analysis.

Furthermore, we conducted a comparative analysis in Table 22. This analysis showed that the results yielded by the methodologies presented in this paper are superior to existing frameworks. For instance, in the case of binary classification, the TAC obtained by the GA-RF-f_3

TABLE 7. Binary classification for f_3.

Model	FV	VAC	TAC	RC	PR	F1S
LR	f_3	86.54 %	74.49 %	86.82 %	72.37 %	78.94 %
DT	f_3	94.74 %	86.53 %	95.96	82.46 %	88.70 %
RF	f_3	95.87 %	87.61 %	98.34 %	82.51 %	89.73 %
ET	f_3	95.72 %	87.38 %	97.86 %	82.48 %	89.51 %
XGB	f_3	94.87 %	86.84 %	98.64 %	81.40 %	89.19 %

TABLE 8. Binary classification for f_4.

Model	FV	VAC	TAC	RC	PR	F1S
LR	f_4	88.12 %	75.51 %	94.26 %	70.87 %	80.91 %
DT	f_4	94.78 %	85.53 %	94.45 %	81.99 %	87.78 %
RF	f_4	95.93 %	86.19 %	96.85 %	81.53 %	88.53 %
ET	f_4	95.85 %	86.13 %	96.62 %	81.59 %	88.47 %
XGB	f_4	94.63 %	85.17 %	96.76 %	80.33 %	87.78 %

TABLE 9. Binary classification for f_5.

Model	FV	VAC	TAC	RC	PR	F1S
LR	f_5	90.01 %	71.94 %	87.10 %	69.59 %	77.37 %
DT	f_5	94.64 %	85.90 %	95.83 %	81.72 %	88.21 %
RF	f_5	96.02 %	86.92 %	98.57 %	81.54 %	89.25 %
ET	f_5	95.96 %	86.60 %	98.35 %	81.26 %	88.99 %
XGB	f_5	94.92 %	86.44 %	98.29 %	81.09 %	88.87 %

TABLE 10. Binary classification for f_6.

Model	FV	VAC	TAC	RC	PR	F1S
LR	f_6	87.63 %	74.49 %	90.21 %	71.17 %	79.56 %
DT	f_6	94.78 %	83.03 %	89.43 %	81.54 %	85.30 %
RF	f_6	95.89 %	87.24 %	98.33 %	82.05 %	89.46 %
ET	f_6	96.05 %	86.99 %	97.89 %	81.98 %	89.23 %
XGB	f_6	94.93 %	86.69 %	98.14 %	81.47 %	89.03 %

TABLE 11. Binary classification for f_7.

Model	FV	VAC	TAC	RC	PR	F1S
LR	f_7	92.12 %	76.81 %	91.57 %	73.10 %	81.30 %
DT	f_7	94.74 %	86.42 %	96.26 %	82.15 %	88.64 %
RF	f_7	96.11 %	87.14 %	98.62 %	81.77 %	89.41 %
ET	f_7	96.11 %	86.85 %	98.26 %	81.61 %	89.16 %
XGB	f_7	94.99 %	86.52 %	98.27 %	81.19 %	88.92 %

TABLE 12. Binary classification for f_8.

Model	FV	VAC	TAC	RC	PR	F1S
LR	f_8	91.71 %	77.71 %	98.46 %	71.65 %	82.94 %
DT	f_8	94.87 %	85.41 %	94.22 %	81.98 %	87.67 %
RF	f_8	95.88 %	87.28 %	98.59 %	81.96 %	89.51 %
ET	f_8	95.80 %	86.92 %	98.12 %	81.77 %	89.20 %
XGB	f_8	94.89 %	86.27 %	98.44 %	80.81 %	88.76 %

TABLE 13. Binary classification for f_9.

Model	FV	VAC	TAC	RC	PR	F1S
LR	f_9	90.02 %	70.83 %	85.92 %	68.83 %	76.43 %
DT	f_9	94.78 %	83.33 %	89.83 %	81.71 %	85.58 %
RF	f_9	95.76 %	87.31 %	98.55 %	82.03 %	89.53 %
ET	f_9	95.75 %	87.14 %	98.54 %	81.82 %	89.41 %
XGB	f_9	94.65 %	86.87 %	99.03 %	81.23 %	89.25 %

TABLE 14. Binary classification for f_{10}.

Model	FV	VAC	TAC	RC	PR	F1S
LR	f_{10}	86.92 %	72.70 %	82.71 %	71.92 %	76.94 %
DT	f_{10}	94.79 %	86.51 %	96.29 %	82.24 %	88.71 %
RF	f_{10}	95.86 %	86.71 %	98.57 %	81.27 %	89.09 %
ET	f_{10}	95.70 %	86.32 %	98.29 %	80.95 %	88.78 %
XGB	f_{10}	94.87 %	86.41 %	99.13 %	80.63 %	88.93 %

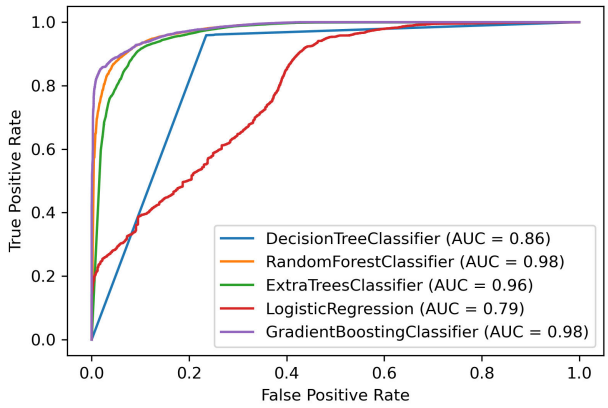

FIGURE 4. ROC Curves for classifiers using f_3.

TABLE 15. Multiclass classification for g_1.

Model	FV	VAC	TAC	PR	RC	F1S
DT	g_1	81.05	74.56	80.64	74.56	77.48
RF	g_1	82.84	76.61	82.85	76.61	79.61
ET	g_1	82.90	76.53	82.63	76.53	79.46
XGB	g_1	82.95	76.48	82.62	76.48	79.43
NB	g_1	54.46	52.28	59.90	52.29	55.83

TABLE 16. Multiclass classification for g_2.

Model	FV	VAC	TAC	PR	RC	F1S
DT	g_2	81.26	75.08	80.36	75.08	77.63
RF	g_2	82.96	77.18	82.70	77.18	79.84
ET	g_2	83.07	77.24	82.48	77.24	79.77
XGB	g_2	83.03	77.23	82.46	77.23	79.76
NB	g_2	55.10	51.59	69.39	51.59	59.18

(proposed in this work) was 11.51% higher than the work presented in [25], 12.1% higher than the method in [32] and 3.71% greater than TAC obtained in [26]. In the case of the multiclass classification process, the GA-ET-$g5$ obtained a TAC that is 5.11% greater than the TAC obtained in [32] and 1.87% higher than the TAC obtained in [33]. Furthermore, the methods that were proposed in this research were

TABLE 17. Multiclass classification for g_3.

Model	FV	VAC	TAC	PR	RC	F1S
DT	g_3	80.59	74.03	80.64	74.03	77.20
RF	g_3	82.20	76.04	82.43	76.04	79.11
ET	g_3	82.22	76.12	82.10	76.12	79.00
XGB	g_3	82.27	76.11	82.14	76.11	79.01
NB	g_3	46.62	55.47	52.69	55.47	54.05

TABLE 18. Multiclass classification for g_4.

Model	FV	VAC	TAC	PR	RC	F1S
DT	g_4	80.99	73.85	80.54	73.85	77.05
RF	g_4	82.50	76.08	83.29	76.08	79.50
ET	g_4	82.51	76.35	83.35	76.35	79.70
XGB	g_4	82.57	76.41	83.32	76.41	79.72
NB	g_4	35.19	32.72	68.81	32.72	44.35

TABLE 19. Multiclass classification for g_5.

Model	FV	VAC	TAC	PR	RC	F1S
DT	g_5	81.44	75.70	81.09	75.70	78.30
RF	g_5	82.93	77.34	83.01	77.34	80.07
ET	g_5	82.94	77.64	83.09	77.64	80.27
XGB	g_5	82.94	77.58	82.99	77.58	80.20
NB	g_5	43.40	40.26	66.38	40.26	50.13

TABLE 20. Multiclass classification for g_6.

Model	FV	VAC	TAC	PR	RC	F1S
DT	g_6	80.57	74.40	74.40	80.15	77.16
RF	g_6	82.52	76.41	82.37	76.41	79.28
ET	g_6	82.64	76.56	82.43	76.56	79.39
XGB	g_6	82.67	76.54	82.32	76.54	79.33
NB	g_6	50.49	47.32	75.32	47.32	58.12

TABLE 21. Multiclass classification for g_7.

Model	FV	VAC	TAC	PR	RC	F1S
DT	g_7	81.38	74.90	74.90	80.60	77.65
RF	g_7	82.65	76.86	82.93	76.86	79.78
ET	g_7	82.87	76.86	82.83	76.86	79.73
XGB	g_7	82.83	76.84	82.86	76.84	79.74
NB	g_7	47.56	43.55	67.91	43.55	53.07

superior to the DL-based algorithms that were reviewed in the literature. For instance, the GA-RF achieved a TAC that is 2.19% higher than the TAC obtained by the LSTM method in [35]. In comparison to the TAC obtained by the LSTM approach in [36], the GA-RF attained a TAC that is 6.89% higher. Additionally, the GA-RF achieved a higher TAC in comparison to the CNN-RNN presented in [37]. Additionally, the GA-RF presented in this paper achieved an accuracy that is superior to existing research. For instance, for the two-way classification task, it achieved a TAC that is 0.9% higher than the performance obtained by the GA-RF in [38]. For the multiclass classification procedure, it obtained an accuracy that is 13.34% higher than the score obtained by the GA-RF in [38].

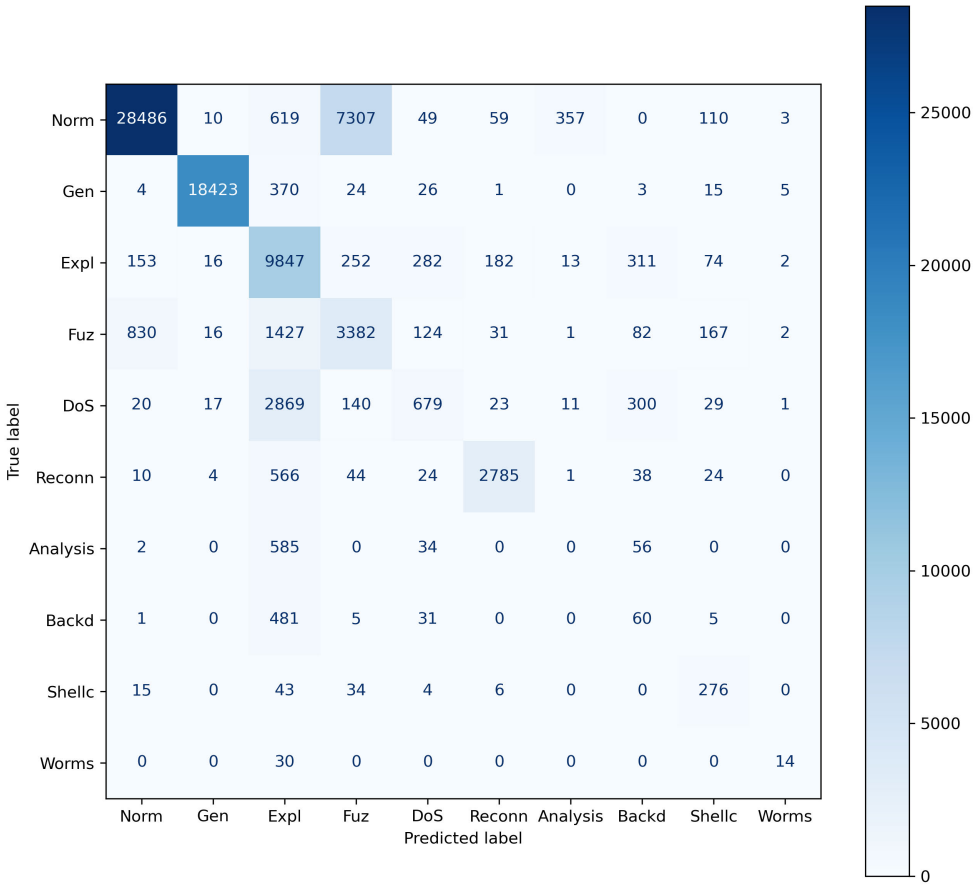

FIGURE 5. Confusion Matrix for g_5 results.

TABLE 22. Comparison with other methods.

Model	TAC- Binary	TAC- Multiclass
PSO-LightGBM [22]	86.68%	-
APAC-IELM [24]	-	70.52%
Deep learning - DNN [25]	76.1%	65.1%
ANN [26]	83.9%	-
GA-SVM [28]	86.38%	-
GWO-SVM [28]	84.48%	-
FFA-SVM [28]	85.42%	-
IG-TS [30]	85.78%	-
GA-LR-DT [31]	81.42%	-
XGBoost-LR [32]	75.51%	72.53%
SVM-NIDS [33]	85.99%	75.77%
IG-Tree [34]	84.83%	-
Deep learning - LSTM [35]	85.42%	-
Deep learning - LSTM [36]	80.72%	72.26%
Deep learning - CNN-RNN [37]	86.64%	-
GA - RF [38]	86.70%	-
GA - RF [39]	-	64.23%
GA - RF (Proposed)	87.61%	-
GA - ET (Proposed)	-	77.64%

Moreover, a performance analysis of prediction time was conducted between different models that used the most optimal feature vectors. In the instance of the binary classification, the vector that yielded the most optimal TAC is f_3.

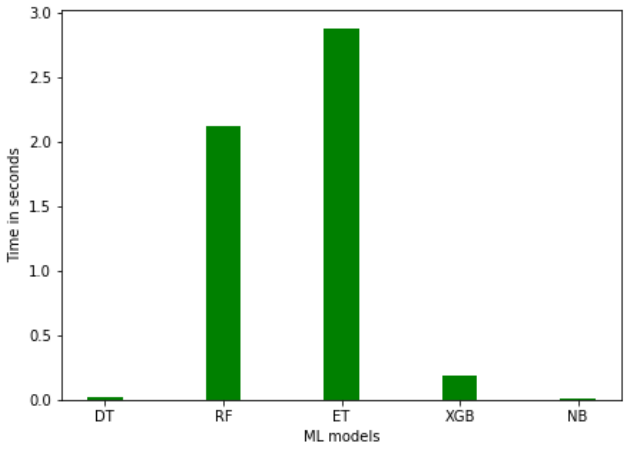

FIGURE 6. Prediction time - Binary classification - f_3.

The graph in Figure 6 shows that the DT model is the most efficient method in terms of prediction time (18.3 milliseconds) when using f_3. For the multiclass classification process, the vector that achieved the highest TAC is g_5. The plot in Figure 7 demonstrates that the NB (7.96 milliseconds) method was the most efficient one in terms of prediction

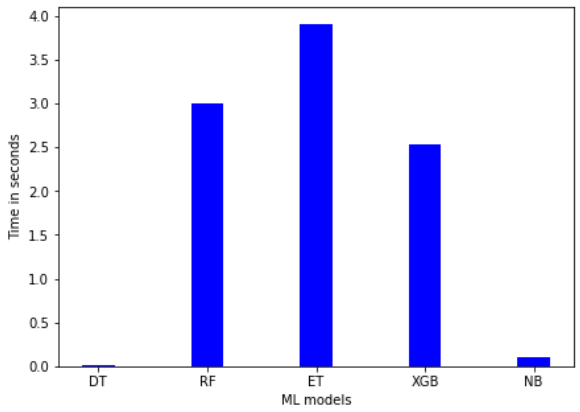

FIGURE 7. Prediction time - Multiclass classification - g_5.

time when utilizing g_5. However, the NB did not obtain a satisfactory TAC.

VI. CONCLUSION

In this research, an advanced IDS system for IIoT was proposed and it was evaluated using the UNSW-NB15 dataset. This IDS was designed using multiple stages. The first stage involved implementing the GA algorithm in conjunction with the RF model to select the most important features to be used by the classifiers. This stage generated two sets of feature vectors. The first feature set, V_b, included 10 feature vectors destined for the binary classification procedure. The second feature set, V_m, contained 7 feature vectors that were used for the multiclass modeling process. For the binary classification experiments, the LR algorithm was applied as the baseline model and the following Tree-based models were implemented: DT, RF, ET, and XGB. For the multiclass modeling process, the NB was used as the baseline model alongside the same Tree-based algorithms that were implemented for the binary intrusion detection procedure. The results demonstrated that for the binary classification process, the GA-RF achieved a TAC of 87.61% and an AUC of 0.98 using f_3 that contained 16 features. When modeling for the multiclass classification, the outcomes showed that the GA-ET got a TAC of 77.64% using g_5 that contained 17 attributes. The results achieved by the methods proposed in this study were superior in comparison to those achieved by the existing methodologies. In future work, we intend to pair the GA algorithm with models such as the SVM or ANN. We also aim to increase the performance of our proposed approach on the minority classes of the UNSW-NB15. Furthermore, we intend to implement the proposed methodology on the TON_IoT. This dataset contains traffic patterns that have been mainly generated by IIoT devices. Additionally, we intend to conduct a performance analysis of the method proposed in this paper across multiple datasets including the NSL-KDD and the AWID.

REFERENCES

[1] Y. Zhang, P. Li, and X. Wang, "Intrusion detection for IoT based on improved genetic algorithm and deep belief network," *IEEE Access*, vol. 7, pp. 31711–31722, 2019.

[2] A. S. Lalos, A. P. Kalogeras, C. Koulamas, C. Tselios, C. Alexakos, and D. Serpanos, "Secure and safe IIoT systems via machine and deep learning approaches," in *Security and Quality in Cyber-Physical Systems Engineering*. Cham, Switzerland: Springer, 2019, pp. 443–470.

[3] B. Valeske, A. Osman, F. Römer, and R. Tschuncky, "Next generation NDE sensor systems as IIoT elements of industry 4.0," *Res. Nondestruct. Eval.*, vol. 31, nos. 5–6, pp. 340–369, Nov. 2020.

[4] R. Schiekofer, A. Scholz, and M. Weyrich, "REST based OPC UA for the IIoT," in *Proc. IEEE 23rd Int. Conf. Emerg. Technol. Factory Autom. (ETFA)*, Sep. 2018, pp. 274–281.

[5] A. Meddeb, "Internet of Things standards: Who stands out from the crowd?" *IEEE Commun. Mag.*, vol. 54, no. 7, pp. 40–47, Jul. 2016.

[6] N. Koroniotis, N. Moustafa, and E. Sitnikova, "A new network forensic framework based on deep learning for Internet of Things networks: A particle deep framework," *Future Gener. Comput. Syst.*, vol. 110, pp. 91–106, Sep. 2020.

[7] A. Khraisat, I. Gondal, P. Vamplew, and J. Kamruzzaman, "Survey of intrusion detection systems: Techniques, datasets and challenges," *Cybersecurity*, vol. 2, no. 1, pp. 1–22, Dec. 2019.

[8] S. Dua and X. Du, *Data Mining and Machine Learning in Cybersecurity*. Boca Raton, FL, USA: CRC Press, 2016.

[9] X. D. Zhang, "Machine learning," in *A Matrix Algebra Approach to Artificial Intelligence*. Singapore: Springer, 2020, pp. 223–440.

[10] M. Mohammed, M. B. Khan, and E. B. M. Bashier, *Machine Learning: Algorithms and Applications*. Boca Raton, FL, USA: CRC Press, 2016.

[11] 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.

[12] H. Gharaee and H. Hosseinvand, "A new feature selection IDS based on genetic algorithm and SVM," in *Proc. 8th Int. Symp. Telecommun. (IST)*, Sep. 2016, pp. 139–144.

[13] R. Wald, T. M. Khoshgoftaar, and A. Napolitano, "Stability of filter- and wrapper-based feature subset selection," in *Proc. IEEE 25th Int. Conf. Tools With Artif. Intell.*, Nov. 2013, pp. 374–380.

[14] M. Shafiq, Z. Tian, A. K. Bashir, X. Du, and M. Guizani, "IoT malicious traffic identification using wrapper-based feature selection mechanisms," *Comput. Secur.*, vol. 94, Jul. 2020, Art. no. 101863.

[15] M. A. Siddiqi and W. Pak, "Optimizing filter-based feature selection method flow for intrusion detection system," *Electronics*, vol. 9, no. 12, p. 2114, Dec. 2020.

[16] S. Ding, X. Xu, H. Zhu, J. Wang, and F. Jin, "Studies on optimization algorithms for some artificial neural networks based on genetic algorithm (GA)," *J. Comput.*, vol. 6, no. 5, pp. 939–946, May 2011.

[17] P. Probst, M. N. Wright, and A. Boulesteix, "Hyperparameters and tuning strategies for random forest," *Wiley Interdiscipl. Rev., Data Mining Knowl. Discovery*, vol. 9, no. 3, May 2019, Art. no. e1301.

[18] V. Kumar, A. K. Das, and D. Sinha, "Statistical analysis of the UNSW-NB15 dataset for intrusion detection," in *Computational Intelligence in Pattern Recognition*. Singapore: Springer, 2020, pp. 279–294.

[19] N. Moustafa and J. Slay, "UNSW-NB15: A comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set)," in *Proc. Mil. Commun. Inf. Syst. Conf. (MilCIS)*, Nov. 2015, pp. 1–6.

[20] N. Moustafa and J. Slay, "The evaluation of network anomaly detection systems: Statistical analysis of the UNSW-NB15 data set and the comparison with the KDD99 data set," *Inf. Secur. J., Global Perspective*, vol. 25, nos. 1–3, pp. 18–31, 2016.

[21] A. Alsaedi, N. Moustafa, Z. Tari, A. Mahmood, and A. Anwar, "TON_IoT telemetry dataset: A new generation dataset of IoT and IIoT for data-driven intrusion detection systems," *IEEE Access*, vol. 8, pp. 165130–165150, 2020.

[22] J. Liu, D. Yang, M. Lian, and M. Li, "Research on intrusion detection based on particle swarm optimization in IoT," *IEEE Access*, vol. 9, pp. 38254–38268, 2021.

[23] X. Zhou, Y. Hu, W. Liang, J. Ma, and Q. Jin, "Variational LSTM enhanced anomaly detection for industrial big data," *IEEE Trans. Ind. Informat.*, vol. 17, no. 5, pp. 3469–3477, May 2021.

[24] J. Gao, S. Chai, B. Zhang, and Y. Xia, "Research on network intrusion detection based on incremental extreme learning machine and adaptive principal component analysis," *Energies*, vol. 12, no. 7, p. 1223, Mar. 2019.

[25] R. Vinayakumar, M. Alazab, K. Soman, P. Poornachandran, A. Al-Nemrat, and S. Venkatraman, "Deep learning approach for intelligent intrusion detection system," *IEEE Access*, vol. 7, pp. 41525–41550, 2019.

[26] S. Hanif, T. Ilyas, and M. Zeeshan, "Intrusion detection in IoT using artificial neural networks on UNSW-15 dataset," in *Proc. IEEE 16th Int. Conf. Smart Cities, Improving Qual. Life Using ICT IoT AI (HONET-ICT)*, Oct. 2019, pp. 152–156.
[27] E. Ketzaki, A. Drosou, S. Papadopoulos, and D. Tzovaras, "A light-weighted ANN architecture for the classification of cyber-threats in modern communication networks," in *Proc. 10th Int. Conf. Netw. Future (NoF)*, Oct. 2019, pp. 17–24.
[28] O. Almomani, "A feature selection model for network intrusion detection system based on PSO, GWO, FFA and GA algorithms," *Symmetry*, vol. 12, no. 6, p. 1046, Jun. 2020.
[29] A. Nazir and R. A. Khan, "A novel combinatorial optimization based feature selection method for network intrusion detection," *Comput. Secur.*, vol. 102, Mar. 2021, Art. no. 102164.
[30] W. Zong, Y.-W. Chow, andW. Susilo, "A two-stage classifier approach for network intrusion detection," in *Proc. Int. Conf. Inf. Secur. Pract. Exper.* Cham, Switzerland: Springer, 2018, pp. 329–340.
[31] C. Khammassi and S. Krichen, "A GA-LR wrapper approach for feature selection in network intrusion detection," *Comput. Secur.*, vol. 70, pp. 255–277, Sep. 2017.
[32] S. M. Kasongo and Y. Sun, "Performance analysis of intrusion detection systems using a feature selection method on the UNSW-NB15 dataset," *J. Big Data*, vol. 7, no. 1, pp. 1–20, Dec. 2020.
[33] D. Jing and H.-B. Chen, "SVM based network intrusion detection for the UNSW-NB15 dataset," in *Proc. IEEE 13th Int. Conf. ASIC (ASICON)*, Oct. 2019, pp. 1–4.
[34] V. Kumar, D. Sinha, A. K. Das, S. C. Pandey, and R. T. Goswami, "An integrated rule based intrusion detection system: Analysis on UNSW-NB15 data set and the real time online dataset," *Cluster Comput.*, vol. 23, no. 2, pp. 1397–1418, Jun. 2020.
[35] A. Aleesa, M. Younis, A. A. Mohammed, and N. Sahar, "Deep Intrusion detection system with enhanced UNSW-NB15 dataset based on deep learning techniques," *J. Eng. Sc. Techn.*, vol. 16, no. 1, pp. 711–727, 2021.
[36] A. V. Elijah, A. Abdullah, N. Jhanjhi, M. Supramaniam, and B. Abdullateef, "Ensemble and deep-learning methods for two-class and multi-attack anomaly intrusion detection: An empirical study," *Int. J. Adv. Comput. Sci. Appl.*, vol. 10, no. 9, pp. 520–528, 2019.
[37] P. Wu, H. Guo, and N. Moustafa, "Pelican: A deep residual network for network intrusion detection," in *Proc. 50th Annu. IEEE/IFIP Int. Conf. Dependable Syst. Netw. Workshops (DSN-W)*, Jun. 2020, pp. 55–62.
[38] A. Assiri, "Anomaly classification using genetic algorithm-based random forest model for network attack detection," *Comput., Mater. Continua*, vol. 66, no. 1, pp. 767–778, 2020.
[39] C. Khammassi and S. Krichen, "A NSGA2-LR wrapper approach for feature selection in network intrusion detection," *Comput. Netw.*, vol. 172, May 2020, Art. no. 107183.
[40] W. Li and Z. Liu, "A method of SVM with normalization in intrusion detection," *Procedia Environ. Sci.*, vol. 11, pp. 256–262, Jan. 2011.
[41] H. Sharma and S. Kumar, "A survey on decision tree algorithms of classification in data mining," *J. Sci. Res.*, vol. 5, no. 4, pp. 2094–2097, 2016.
[42] J. Liang, Z. Qin, S. Xiao, L. Ou, and X. Lin, "Efficient and secure decision tree classification for cloud-assisted online diagnosis services," *IEEE Trans. Depend. Sec. Comput.*, vol. 18, no. 4, pp. 1632–1644, Jul. 2021.
[43] L. Breiman, "Random forests," *Mach. Learn.*, vol. 45, no. 1, pp. 5–32, 2001.
[44] A. Liaw and M. Wiener, "Classification and regression by randomforest," *R News*, vol. 2, no. 3, pp. 18–22, 2002.
[45] G. Biau and E. Scornet, "A random forest guided tour," *Test*, vol. 25, no. 2, pp. 197–227, 2016.
[46] P. Geurts, D. Ernst, and L. Wehenkel, "Extremely randomized trees," *Mach. Learn.*, vol. 63, no. 1, pp. 3–42, 2006.
[47] L. Davis, *Handbook of Genetic Algorithms*. New York, NY, USA: Van Nostrand Reinhold, 1991.
[48] X. Yu and M. Gen, *Introduction to Evolutionary Algorithms*. London, U.K.: Springer, 2010.
[49] P. Tao, Z. Sun, and Z. Sun, "An improved intrusion detection algorithm based on GA and SVM," *IEEE Access*, vol. 6, pp. 13624–13631, 2018.
[50] M. Almseidin, M. Alzubi, S. Kovacs, and M. Alkasassbeh, "Evaluation of machine learning algorithms for intrusion detection system," in *Proc. IEEE 15th Int. Symp. Intell. Syst. Informat. (SISY)*, Sep. 2017, pp. 000277–000282.
[51] S. Narkhede. *Understanding AUC-ROC Curve*. Towards Data Science. Accessed: Apr. 4, 2021. [Online]. Available: https://towardsdatascience.com/understanding-auc-roc-curve-68b2303cc9c5
[52] Scikit-Learn. *Machine Learning in Python*. Accessed: May 3, 2020. [Online]. Available: https://scikit-learn.org/stable/
[53] A. De Caigny, K. Coussement, and K. W. De Bock, "A new hybrid classification algorithm for customer churn prediction based on logistic regression and decision trees," *Eur. J. Oper. Res.*, vol. 269, pp. 760–772, Sep. 2018.
[54] M. M. Saritas, "Performance analysis of ANN and naive Bayes classification algorithm for data classification," *Int. J. Intell. Syst. Appl. Eng.*, vol. 7, no. 2, pp. 88–91, Jan. 2019.

SYDNEY MAMBWE KASONGO received the bachelor's and master's degrees *(cum laude)* in computer systems from Tshwane University of Technology (TUT), in 2015 and 2017, respectively, and the Ph.D. degree in electrical and electronic engineering from the University of Johannesburg (UJ), focusing on deep learning applied to intrusion detection systems. He is currently a Data Science Lecturer with the Department of Industrial Engineering and the School for Data Science and Computational Thinking, Stellenbosch University, South Africa.