A Dependable Hybrid Machine Learning Model for Network Intrusion Detection

Md. Alamin Talukder[a], Khondokar Fida Hasan[b], Md. Manowarul Islam[a], Md Ashraf Uddin[a], Arnisha Akhter[a], Mohammad Abu Yousuf[c], Fares Alharbi[d], Mohammad Ali Moni[e]

[a]*Department of Computer Science and Engineering, Jagannath University, Dhaka, Bangladesh*
[b]*Information Security Discipline, School of Computer Science, Queensland University of Technology (QUT), 2 George Street, Brisbane 4000, Australia*
[c]*Institute of Information Technology, Jahangirnagar University, Savar, Dhaka, Bangladesh*
[d] *Computer Science Department, Shaqra University, Shaqra 15526, Saudi Arabia*
[e]*Artificial Intelligence & Data Science, School of Health and Rehabilitation Sciences, Faculty of Health and Behavioural Sciences, The University of Queensland St Lucia, QLD 4072, Australia.*

Abstract

Network intrusion detection systems (NIDSs) play an important role in computer network security. There are several detection mechanisms where anomaly-based automated detection outperforms others significantly. Amid the sophistication and growing number of attacks, dealing with large amounts of data is a recognized issue in the development of anomaly-based NIDS. However, do current models meet the needs of today's networks in terms of required accuracy and dependability? In this research, we propose a new hybrid model that combines machine learning and deep learning to increase detection rates while securing dependability. Our proposed method ensures efficient pre-processing by combining SMOTE for data balancing and XGBoost for feature selection. We compared our developed method to various machine learning and deep learning algorithms in order to find a more efficient algorithm to implement in the pipeline. Furthermore, we chose the most effective model for network intrusion based on a set of benchmarked performance analysis criteria. Our method produces excellent results when tested on two datasets, KDDCUP'99 and CIC-MalMem-2022, with an accuracy of 99.99% and 100% for KDDCUP'99 and CIC-MalMem-2022, respectively, and no overfitting or Type-1 and Type-2 issues.

Keywords: Intrusion Detection System, Machine Learning, XGBoost, Feature Selection, Feature Importance, Accuracy, Dependability

1. Introduction

Internet-based computer networks are becoming more vulnerable to security threats. The constant emergence of new types of threats makes developing dependable and adaptable security strategies a critical issue. Important data is always a target for attackers, making it vulnerable to concentrated network attacks. The process by which an attacker gains access to a system or system server and then sends malicious packets to the user system in order to steal, modify, or corrupt any sensitive or vital data is referred to as intrusion. An attack is defined as the unauthorized transmission of network packets or malicious conduct over a network. Existing system vulnerabilities,

*Mohammad Ali Moni
**Md Manowarul Islam
Email addresses: mdalamintalukdercsejnu@gmail.com (Md. Alamin Talukder), fida.hasan@qut.edu.au (Khondokar Fida Hasan), manowar@cse.jnu.ac.bd (Md. Manowarul Islam), ashraf@cse.jnu.ac.bd (Md Ashraf Uddin), arnisha@cse.jnu.ac.bd (Arnisha Akhter), yousuf@juniv.edu (Mohammad Abu Yousuf), faalhrbi@su.edu.sa (Fares Alharbi), m.moni@uq.edu.au (Mohammad Ali Moni)

such as user error, misconfiguration, or software flaws, may allow the intrusion to occur on the server or system. An intelligent intrusion can also be carried out by combining various system vulnerabilities.

In a global network, a vast number of online services and millions of massive servers are active. As a result, as these networks become more appealing to attackers, they require intrusion detection systems (IDS) to secure their network systems. According to the 2020 Trustwave Global Security Report, phishing and social engineering attacks accounted for 50% of incidents that affected corporate networks, e-commerce (22%) systems, cloud (20%) systems, and point-of-sale (5%) systems. 86% of intrusion detection occurrences in 2019 were related to external events or internet usage [1].

IDS are typically classified into different categories based on the user's perspective. For example, IDS can be host-based (HIDS) or network-based (NIDS) [2]. NIDS detects infiltration by monitoring network traffic and hosts connecting to the network. HIDS examines device calls, file system modifications, application logs, and host activity to detect an intrusion.

On the other hand, the type of analysis performed determines whether an intrusion detection system is signature-based (SIDS) or anomaly-based (NIDS). Signature-based schemes (also known as misuse-based schemes) look for predefined patterns called signatures in the analyzed data. For this reason, a prior signature database corresponding to known assaults is specified. On the other hand, anomaly-based detectors attempt to estimate the protected system's "normal" behavior and emit an anomaly warning whenever the difference between a given observation at a given instant and the typical behavior exceeds a predetermined threshold. Another option is to model the system's "abnormal" behavior and set off an alarm when the difference between observed and expected behavior falls below a certain threshold. SIDS detects intrusions based on specific patterns in malware transmission, such as the number of bytes and the order of malicious blueprints. The most recent malware attack patterns, on the other hand, are frequently unknown and unpredictable. As a result, SIDS models trained on out-of-date datasets are unable to detect new malware attacks. AIDS, on the other hand, detects unknown malware by comparing incoming packets to the model's previous knowledge and classifying them as suspicious or not. Machine learning has become much more popular for developing models. However, it is becoming increasingly difficult to accommodate large amounts of additional information [3, 4] in many fields, including machine learning, data analysis, and text mining [5, 6].

Machine Learning is an effective method for obtaining pattern information from well-defined inputs without depending on an algorithmic method to develop meaningful outputs [7, 8]. Machine Learning-based intrusion detection systems aid in detecting anomalous behavior of internet-connected devices and are gaining popularity. Several processes and methods using machine learning are already presented for detecting network intrusion. And due to their superior ability to detect intrusions and perform generalizations, effective IDS are typically created nowadays by utilizing machine learning-based approaches. However, the implementation of such systems is inherently difficult. The system's intrinsic challenges can be divided into different problem groups based on competence, accuracy, and dependability. For example, in comparison with earlier detection methods that relied on malicious signatures, machine learning-based IDS, and particularly those strategies that are based on anomaly detection, often show a larger percentage of false-positive occurrences. In addition, compared to existing approaches, the system's learning process takes a significant amount of training data and is quite sophisticated. Another issue is the irrelevant and excessive information that increases the complexity of the dimension and hinders accurate classification predictions,

which negatively affects the performance of the system [9, 10]. Such impact of big data on dimensionality causes poor performance, a high frequency of false positives, and other issues that reduce the performance and reliability of the model [11]. Overall, new and evolving threats to these applications are always appearing, necessitating the need for reliable and advanced protection solutions. The classic machine learning-based IDS must be updated to meet the security needs of the present sustainable environment due to the quick proliferation of the network's security needs and changing threat kinds.

This paper presents a dependable IDS model that outperforms a number of current methods using a hybrid approach utilizing machine learning (ML) and deep learning (DL) algorithms. In this hybrid technique, a deep learning-based feature selection has been used to reduce dimensionality. By eliminating superfluous features and filtering out unnecessary data, it reduces the complexity of the data and its dimensions. Additionally, it helps to lessen the workload associated with computing while simultaneously enhancing detecting capabilities [12]. Additionally, to balance the dataset and improve the effectiveness of minority incursions, which are motivated by a significant class imbalance in the intrusion dataset [13, 14], synthetic minority oversampling technology (SMOTE) [15] is utilized. Overall, the suggested model makes use of dimensionality reduction, feature selection, and feature extraction to compress the extracted information. In order to reduce the dimension of imbalanced datasets, we use pre-processing techniques in our study to develop a productive and embodied architecture. Normalization, data balancing, and dimension reduction are necessary for an unbalanced dataset. In order to achieve our computations in less time with adequate prediction performance for both binary and multilabel classifications, SMOTE is used to balance our datasets, and XGBoost is used as a feature selection strategy to lower the dimension.

A variety of ML classifiers, including Random Forest (RF), Decision Tree (DT), K-Nearest Neighbor (KNN), Multilayer Perceptron (MLP), Convolution Neural Network (CNN), and Artificial Neural Network (ANN), were used to analyze the performance of our proposed model. For each classifier's training, a small subset of the dataset's most crucial features was chosen using the XGBoost algorithm. The performance parameters used to assess the model include accuracy, precision, recall, f1-score, AUC score, ROC Curve, MAE, MSE, and RMSE. The performance analysis showed that the proposed model could detect attacks with an accuracy above 99.9% for all ML classifier algorithms.

The overall contributions of the paper are summarized as follows:

- We proposed a novel hybrid approach and showed how reliable it is for detecting network intrusion by interpreting the dependability in metrics of accuracy, availability, and scalability of the model.
- XGBoost for feature selection, SMOTE for data balancing, and proper preprocessing, we developed our own hybrid approach that uniquely outperforms the state-of-the-art models for network intrusion detection.
- Finally, we used a number of performance indicators to assess how well the model can perform; the results show that our hybrid model is superior when compared with the existing model in detecting intrusions, resulting in lower type-1 (False Positive) and type-2 (False Negative) rates.

The succeeding units provide a summary of the existing work, our proposed model, and the results. Initially, the related work is presented in Section 2. The proposed methodology is described in Section 3, followed by presenting

the details of our datasets, experiments, and findings in this paper in Section 4. Finally, the conclusion and proposed future work are presented in Section 5.

2. Related Works

In recent years, a number of intrusion detection and prevention techniques have been proposed. To solve the problem of class imbalance as well as improve intrusion detection efficacy. In [13], the authors developed a strategy for balancing the data while using the SMOTE, subsequently training with the RF algorithm. The experiments were performed on a benchmark KDDCUP'99 intrusion dataset, and the RF algorithm's accuracy was 92.39%, which was greater than that of other comparable techniques. The RF combined with the SMOTE was used to have an accuracy of 92.57% upon resampling the minority samples.

To increase the effectiveness and correctness of IDS, the authors [16] proposed an ensemble-based IDS leveraging XGBoost. They demonstrated that using XGBoost alongside ensemble-based IDS could produce superior performance since XGBoost depends on tree-boost machine learning techniques that aid in a gentler "bias-variance" barter. The study was carried out using the KDDCUP'99 data, and the suggested strategy's accuracy was determined to be 99.95%.

A feature selection strategy for identifying DoS and DDoS attacks that uses insertion and union processes on subsets by the top 50% Information Gain (IG) and Gain Ratio (GR) features [17]. With the help of a JRipclassifier, the suggested technique was tested and validated on the IoT-BoT and KDDCup'99 datasets. On the IoT-BoT and KDDCup'99 datasets, using 16 and 19 features, they outperformed the original feature set and typical IDSs with an accuracy of 99.9993% and 99.992%, respectively.

To recognize the attacks in IoT a Deep Neural Network (DNN) was developed by [18]. The performance of DNN to detect the attacks had been assessed using the three most commonly used datasets such as KDDCUP'99, NSL-KDD, and UNSW-NB15. The proposed method using DNN showed that the accuracy rate was only 91.50% with each dataset. In [19], the particle swarm optimization (PSO) feature selection technique and ANN classifier were utilized to detect intrusion as normal and abnormal activities. They selected 20 features using PSO and built their model to produce better performance and achieved 98.00% accuracy in KDDCUP'99 dataset.

In [20], the authors introduced a method that combines feature correlation (CR) to choose the key features and a DNN classifier to create an IDS model for detecting the intrusion in order to enhance the effectiveness of the network security model. The KDDCUP'99 dataset was utilized in the study, and only 30 attributes were chosen using the CR approach and built the intrusion detection model. With an accuracy rating of 99.40%, their IDS model is capable of detecting intrusion.

An innovative method of intrusion detection proposed by [21] that combines ANN with optimized BAT to improve the detection rate of network attacks. The adopted BAT used to pick the 25 best traits in order to preserve the greatest excellence and remove unimportant attributes from the attack. Utilizing the KDDCUP'99 benchmark dataset as well as the SVM algorithm, the experiment achieved an attack detection rate of 94.12%.

To detect unusual traffic data in a network, the authors [22] has introduced an element identification method based on the Convolutional attention LSTM network model. They extracted the shallow features using CNN and

combined them with the attention-LSTM to gain recognition and classification of different networks' behavior. They performed experiments on the KDDCUP'99 dataset and achieved 98.48% accuracy rate.

In [23], the authors employed several ML algorithms to identify intrusion on the KDDCUP'99 dataset to compare the effectiveness of these classifiers. K-fold cross-validation, where k=10, was applied to separate the training and testing part of the dataset to enhance the performance. They got the highest accuracy rate of 94.00% in DT than other algorithms.

In [24], the authors proposed a misuse-based IDS to protect networks from modern attacks, namely DOS, Exploit, Probe, Generic, etc. The UNSW-NB15 dataset was used to examine the act of numerous classification models such as CART, C5, CHAID, and QUEST and found the accuracy, IDR, and FAR of each model. Feature selection was performed using IG to select 13 features from 47. The accuracy rate for the proposed C5 model was 99.37%.

A network forensic mechanism [25] has been developed based on network flow identifiers to mistrustful road events of botnets. The ML classifiers such as DT C4.5, ARM, ANN, and NB with the UNSW-NB15 dataset were used to perceive botnet attacks. The performance measurement was done on the Weka tool for analyzing the classification accuracy and FAR, where default parameters and 10-fold cross-validation were employed. The accuracy was only 86.45%, 93.23%, 72.73%, 63.97% and FAR was 13.55%, 6.77%, 27.27%, 36.03% for ARM, DT, NB and ANN respectively.

In [26], the authors developed a filter-based feature-dropping method using the XGBoost algorithm and applied ML algorithms such as DT, ANN, KNN, support vector machine (SVM), and LR for accuracy prediction on the UNSW-NB15 IDS dataset. They confirmed that their developed method increased binary classification accuracy from 88.13% to 90.85%. The overall accuracy was only 90.85%, 84.39%, 77.64%, 84.46%, 60.89% for binary and 67.57%, 77.51%, 65.29%, 72.30%, 53.95% for multiclass of DT, ANN, LR, KNN, SVM respectively.

In [27], the authors introduced a hybrid model to reduce dimension, which combines the IG and principal component analysis (PCA) techniques and an ensemble classifier based on SVM, instance-based learning algorithms (IBK), and MLP. The performance was assessed on ISCX 2012, NSL-KDD, and Kyoto 2006+ datasets. They built the model for binary classification and found the DR (99.10%), accuracy rate (99.01%), and lowest FAR (0.01%) in the ISCX 2012 data set, and the accuracy rate was 98.24% and 99.95%; DR was 98.20% and 99.80%; FAR was 0.017% and 0.021% in the both NSL-KDD and Kyoto 2006 respectively.

A feature selection technique based on linear correlation coefficient (FGCC) and cuttlefish algorithm (CFA) method designed by [28] to detect network intrusion using Decision Tree (DT) algorithm. They applied their approach on KDDCUP'99 dataset to evaluate the performance and got an accuracy rate of 95.03%.

In [29], the authors introduced an algorithm for the reduction of the feature based on filter-based algorithms such as the Input Gain Ratio (IGR), Correlation (CR), and ReliefF (ReF). It produced feature subsets based on the average weight for each classifier and an additional Subset Combination Strategy (SCS). The number of features was reduced from 77 to 24 for CIC-IDS2017 and 41 to 12 for the KDDCUP'99 dataset. It produced an accuracy rate of 99.96% with the rule-based classifier Projective Adaptive Resonance Theory (PART) in 133.66 sec for CIC-IDS2017 and the KDDCUP'99 dataset, the accuracy rate was 99.32%, and the required time was 11.22%.

In [30], the authors implemented an IDS scheme based on MapReduce to procure a small and beneficial number

of features from large datasets to enhance the accuracy of detecting anomalies by the intrusion. The popular KDD-CUP'99 was used for performance evaluation, and for classifying normal and abnormal in mobile cloud computing (MCC) activities RF algorithm was used. The adaptive effective feature selection (EFS) was used to minimize the training set and make the input data parallel. They selected 15 features to evaluate the performance of their model, which achieved 93.90% accuracy.

In [31], the authors included an IDS problem solution Method where Particle Swarm Optimization (PSO) as feature selection and Naive Bayes as a KDDCUP'99 dataset classification algorithm. The dataset presented more than 40 features and over four hundred thousand records. PSO has been used to pick 38 features from 40+ features to prevent further calculation or memory consumption. With fewer times and greater accuracy than other features, the accuracy rate reached 99.12%.

In [32], the authors introduced a new redundant penalty-by-feature (RPFMI) mutual information algorithm to choose efficient malware detection features. The KDDCUP'99 and Kyoto 2006+ datasets for intrusion detection were used for the experimentation. They have shown a higher accuracy rate of the proposed algorithm than others. The accuracy rate was 99.77% (DOS), 96.19%(U2R), 91.07%(R2L) for KDDCUP'99 and 97.74% for Kyoto 2006+ dataset.

In [33], the authors created an EMRFT (Enhanced Multi Relational Fuzzy Decision Tree) for categorizing network intrusion relying on genetic optimization. The classifier's efficacy was increased by using the K-Nearest Neighbor approach to fill in missing values in the data and the Fast Correlation-based feature selection technique to minimize the dimensional space. The study was carried out using the KDDCUP'99 dataset, and the findings showed that EMRFT performed better, with a binary classification accuracy rate of 98.27% and a multilabel classification accuracy of 96.56 percent.

In [34], the authors suggested an extra boosting forest (EBF) model that uses a stacked ensemble strategy to combine the extra tree (ET) classifier, gradient boosting (GB) classifier, and RF models, with the goal of accurately identifying malicious traffic. They used two datasets, namely UNSW-NB15 and IoTID20, which have data on local network traffic and IoT-based traffic, respectively. Utilizing PCA to reduce the number of features in each dataset to 30. The findings demonstrate that EBF scored noticeably better and received the maximum accuracy score of 98.5 and 98.4 in the multilabel of four classes for UNSW-NB15 and IoTID20, respectively.

The VolMemLyzer, among the foremost up-to-date memory feature extractors for learning environments, has been modified to emphasize disguised and obfuscated malware and combined with a stacked ensemble machine learning paradigm to develop a framework for quickly recognizing malware [35]. A malware memory dataset (MalMemAnalysis2022) was also constructed to test and assess the framework, with the goal of closely emulating legitimate obfuscated malware. The study revealed that by employing memory feature engineering, the suggested technique could identify obfuscated and disguised malware incredibly quickly, yielding accuracy and F1-Scores of 99.00% and 99.02%, correspondingly.

In [36], the authors performed detection mechanisms utilizing a variety of machine and deep learning techniques in a large data set with memory data. Pyspark was used to conduct this investigation on the Apache Spark big data platform in the Google Colaboratory. On the equitable CIC-MalMem-2022 dataset, tests were conducted. Various machine learning and deep learning methods were used to achieve the identification of malware. The

effectiveness of the employed algorithms has been contrasted, and the outcomes have been assessed using several performance measures. The study's greatest accurate method for detecting malware using memory analysis, the Logistic Regression (LR) technique, had an acceptable accuracy of 99.97%. As a result, memory research data is extremely helpful in identifying malware.

In [37], the authors applied a number of tree-based ensemble ML algorithms to the Portable Executable (PE) malware detection. To show how well the algorithms perform across a range of scenarios, the study uses three open-source datasets, including BODMAS, Kaggle, and CIC-MalMem-2022. According to the test results, all tree-based ensembles worked adequately, and there were no statistically relevant performing variations among the various algorithms. According to this research, the performance rates of the Gradient Boosting Machine (GBM), XGBoost, and RF are higher than those of other tree-based ensemble models. They got the accuracy rate of 99.39%, 99.96%, and 100% for the Kaggle, BODMAS, and CIC-MalMem-2022 datasets, correspondingly.

The literature review is summarised in the following Table 1.

3. Proposed Methodology

3.1. Problem Statement

The intrusion detection system (IDS) is a critical security component for ensuring network usability. In recent years, automated intrusion detection within IDS has advanced significantly through the use of artificial intelligence and how effectively and efficiently ML/DL models can be applied is a recognized approach in terms of innovation in this research area. However, as attackers become more sophisticated, network security has become more challenging nowadays. In addition to that, dealing with large amounts of data has always been a challenge when developing security components.

In this research work, we particularly dealt with the following three problems in AI-enabled automation.

Firstly, we discovered from state-of-the-art research that when working on imbalanced datasets, the majority of the IDS model evaluates their performance in terms of precision, recall, and f1-score. And in most cases, accuracy outperforms precision, recall, and f1-score [17, 31]. In addition to that, most papers did not use a confusion matrix to determine type-1 and type-2 errors, which can be critical in evaluating the detection performance of the model. To address these, first of all, we use SMOTE, which handles the data imbalance problem and ensures that the precision, recall, and f1-score are unaffected. This gives high-performance results, like accuracy. We also evaluate our model by developing a confusion matrix that gives more confidence in and reliability of our model by outlining type-1 and type-2 errors.

Secondly, one of the issues that many authors face is the efficient reduction of the dimension [18, 38, 39, 22]. Dimension reduction can be an important consideration while working with relevant features in removing irrelevant or less important features to reduce time and speed up the process. In our proposed method, we deal with that problem effectively by using XGBoost to reduce dimensions and select the best features to reduce computational costs.

Finally, we addressed the issue of dependability - how dependable is the model? As a result, we conduct dependability analysis using key performance evaluation metrics to validate our model's efficiency, availability, and scalability.

SL.No.	Authors	Dataset	Algorithm	Accuracy (In %)
1	[13]	KDDCUP'99	SMOTE + RF	92.57
2	[16]	KDDCUP'99	XGBoost	99.95
3	[17]	IoT-BoT	IG + GR+ JRipclassifier	99.99
		KDDCUP'99		99.57
		NSL-KDD		91.50
4	[18]	KDDCUP'99	DNN	91.50
		UNSWNB-15		91.50
5	[19]	KDDCUP'99	PSO + ANN	98.00
6	[20]	KDDCUP'99	CR + DNN	99.40
7	[21]	KDDCUP'99	BAT + SVM	94.12
8	[22]	KDD-CUP'99	CNN + LSTM	98.48
9	[23]	KDD-CUP'99	DT	94.00
10	[24]	UNSWNB-15	C5	99.37
11	[25]	UNSWNB-15	DT	86.45
			C4.5	93.23
			ARM	72.73
			ANN	63.97
12	[26]	UNSW-NB15	XGBoost + DT, ANN, LR, KNN, SVM	90.85 (DT), 84.39 (ANN), 77.64 (LR), 84.46 (KNN), 60.89 (SVM) -Binary
				67.57 (DT), 77.51 (ANN), 65.29 (LR), 72.30 (KNN), 53.95 (SVM) -Multilabel
		ISCX 2012		99.01
13	[27]	NSL-KDD	Hybrid Model (IG+PCA+SVM+IBK+MLP)	98.24
		Kyoto 2006		99.95
14	[28]	KDDCUP'99	FGCC+CFA + DT	95.03
15	[29]	CIC-IDS2017	PART	99.95
		KDDCUP'99		99.32
16	[30]	KDDCUP'99	RF	93.90
17	[31]	KDDCUP'99	PSO + NB	99.12
18	[32]	KDDCUP'99	RPFMI	99.77 (DoS), 96.19 (U2R), 91.07 (R2L)
		Kyoto 2006+		97.74
19	[33]	KDDCUP'99	EMRFT	98.27 -Binary
				96.56 -Multilabel
20	[34]	UNSW-NB15	EBF	98.5
		IoTID20		98.4
21	[35]	CIC-MalMem-2022	VolMemLyzer + Stacked Ensemble	99.00
22	[36]	CIC-MalMem-2022	LR	99.97
		Kaggle	GBM	99.39%
23	[37]	BODMAS	XGBoost	99.96%
		CIC-MalMem-2022	RF	100%

Table 1: Performance summary of different proposed works.

3.2. Hybrid Model with Machine and Deep Learning

In addressing the aforementioned research issues, we propose a hybrid approach that combines an efficient pre-processing technique with the handling of missing values, data balancing using SMOTE, feature scaling using standardization, and label encoding to prepare the datasets. XGBoost is then used to select the optimal features to feed into ML and DL algorithms in order to construct the models. We analyzed the performance of multiple ML and DL algorithms, including RF, DT, KNN, MLP, CNN, and ANN, in order to recommend the optimal algorithm for detecting network intrusion.

In this section, we introduce the major building blocks of the proposed hybrid model.

3.2.1. Data Balancing using SMOTE

Data balancing is a process of rebalancing data from imbalanced data. SMOTE is a familiar approach for dealing with unbalanced data [40]. Whenever the class distribution is biased forward into a specific category, unbalanced data issues occur. To overcome the unbalanced dataset issue, the SMOTE strategy produces imitation data to achieve a balance over the minority and majority category sizes [41]. To rebalance the minority and majority categories, a parameter is supplied to the SMOTE technique to establish a specified threshold for simulated samples [42]. SMOTE picks relative entries as well as modifies them one column at a time by adding a random number based on the difference between the neighbouring records. [43] whenever the majority or minority ratio is very high, it is required to oversample the minority class rather than undersample it to equalize the minority class ratio. In our hybrid approach, the SMOTE plays a signification role as the datasets which are imbalanced, like KDDCUP'99, need to balance to overcome the overfitting and less realistic prediction models.

3.2.2. Feature Selection using XGBoost

Feature selection is a method of selecting a subset of the underlying features in order to minimize the feature space to the smallest possible size based on some criteria. Feature extraction is a technique for creating a new set of features that can be utilized alone or in combination [44]. Moreover, it can locate and choose the far more beneficial properties inside data. It's an important stage in the machine learning workflow since it assists in minimizing the fitting problems, reducing adaptation efficiency on the testing data, reducing training duration, and reducing model interpretability [45]. There are three main kinds of feature selection methods: filter-based, wrapper-based, and embedded feature selection [46]. Build-in feature selection is available in the embedded feature selection method, which helps to build a model without applying any additional feature selection method. To choose features, the filter-based feature technique employs assessment criteria, including information analysis as well as distance assessment. The wrapper-based feature selection approach builds a subset of features in a particular way before evaluating feature selection using the findings of classifiers. Using the embedded feature selection approach, certain properties can be dynamically removed within classifier construction, allowing feature selection and classification to be done simultaneously [46]. Some examples of embedded methods are RF, Lasso, XGBoost, and LGBoost algorithms. The detection of IDS can be efficiently solved by using feature selection approaches[47, 48, 49].

In our hybrid approach, we used XGBoost to select the optimal features as we already know, feature selection refers to selecting a particular number of features from all available features. Feature selection is essential to

reduce computation costs and enhance the model's performance [50]. Extreme Gradient Boosting (XGBoost) is a systematic gradient boosting method that utilizes extra precise approximate to determine the best model tree. It uses a number of methods to discover the important features from the dataset, primarily structured data. It involves the following: (i) calculating second-order gradients, i.e., second-sectional loss function derivatives (similar to Newton's), which provides more insight into the development of gradients and how to achieve the lowest possible loss function. To minimize the error of the entire model, while the regular gradient increase uses the loss function of our model base (e.g., the Decision tree), the 2nd derivative of XGBoost is used as an approximation. (ii) Advanced regularization (L1 & L2) improving the generalization of the model and training is so profligate and can be spread parallel across clusters [51].

XGBoost offers a resourceful and effective enactment of the stochastic gradient boosting algorithm to improve accuracy [52]. It provides an inbuilt process to directly get feature importance for feature selection by making a relationship between multiple variables and feature importance. It is made up of decision trees, and the prediction is accomplished by using these trees. To generate trees, XGBoost employs the loss and regularization function [53] to generate the main objective function as follows:

$$L(\phi) = \sum_{i=1}^{n} l(y_i, \hat{y_i}) + \sum_{k=1}^{n} \Omega(f_k) \tag{1}$$

where, l is a distinctive convex loss function measuring the difference between $\hat{y_i}$ (target) and y_i (actual) prediction. Ω (penalizes the complexity of the model) is the regularization function in Equ.(2) which controls the complexity of a model and prevents the model from overfitting.

$$\Omega(f) = \gamma^{T} + \frac{1}{2}\lambda \sum_{j=1}^{T} ||w||^2 \tag{2}$$

where λ is projected to reduce the prediction's sensitivity, T represents the number of terminal nodes,γ represents encouraging pruning, and w represents the leaf weights. To optimize the model of Equ.(1) in an additive manner, we can do as follow:

$$L^{(t)} = \sum_{i=1}^{n} l(y_i, \hat{y_i}^{t-1} + f_t(x_i)) + \Omega(f_t) \tag{3}$$

where, $\hat{y_i}^{t}$ be the $i-th$ instance prediction in $t-th$ iteration and f_t enhance our model.

To enhance the objective, 2^{nd} order approximation can be used as follows:

$$L^{(t)} \simeq \sum_{i=1}^{n} [l(y_i, \hat{y_i}^{t-1}) + g_i f_t(x_i) + \frac{1}{2} h_i f_t^2(x_i)] + \sum_{k=1}^{n} \Omega(f_k) \tag{4}$$

where, $g_i = \partial_{\hat{y}^{(t-1)}} l(y_i, \hat{y_i}^{t-1})$ $h_i = \partial^2_{\hat{y}^{(t-1)}}$ are the loss function of 1st and 2nd order gradient statistics.

By eliminating the constant part of Equ.(4) we can get the abridged objective [54] as follows:

$$\tilde{L}^{(t)} = \sum_{i=1}^{n} [g_i f_t(x_i) + \frac{1}{2}(h_i f_t^2(x_i))] + \Omega(f_t) \tag{5}$$

Table 2 shows the ranking of features of the KDD-CUPP'99 dataset generated from our proposed XGBoost algorithm. Figure 1 shows the important features in the horizontal bar chart of the KDD-CUPP'99 dataset generated from our proposed XGBoost algorithm.

Dataset	Features with Ranking
KDDCUP'99 (Binary)	22, 12, 25, 7, 38, 5, 37, 9, 36, 34, 1, 2, 4, 35, 3, 28, 39, 32, 31, 40, 33, 0, 26, 15, 11, 16, 23, 29, 24, 30, 13, 18, 27, 21, 6, 8, 17, 10, 19, 14, 20
KDDCUP'99 (Multilabel)	21, 10, 22, 23, 2, 1, 4, 37, 32, 13, 5, 34, 36, 14, 29, 16, 9, 35, 3, 0, 7, 30, 18, 31, 11, 39, 28, 40, 12, 8, 15, 33, 25, 26, 38, 17, 24, 27, 19, 6, 20
CIC-MalMem-2022	45, 48, 7, 8, 52, 27, 12, 30, 11, 0, 46, 5, 32, 23, 42, 19, 4, 15, 51, 14, 37, 43, 6, 2, 1, 20, 24, 41, 29, 39, 13, 21, 17, 26, 28, 18, 16, 10, 9, 50, 49, 3, 44, 22, 47, 25, 53, 31, 33, 34, 35, 36, 38, 40, 54

Table 2: XGBoost-based feature ranking.

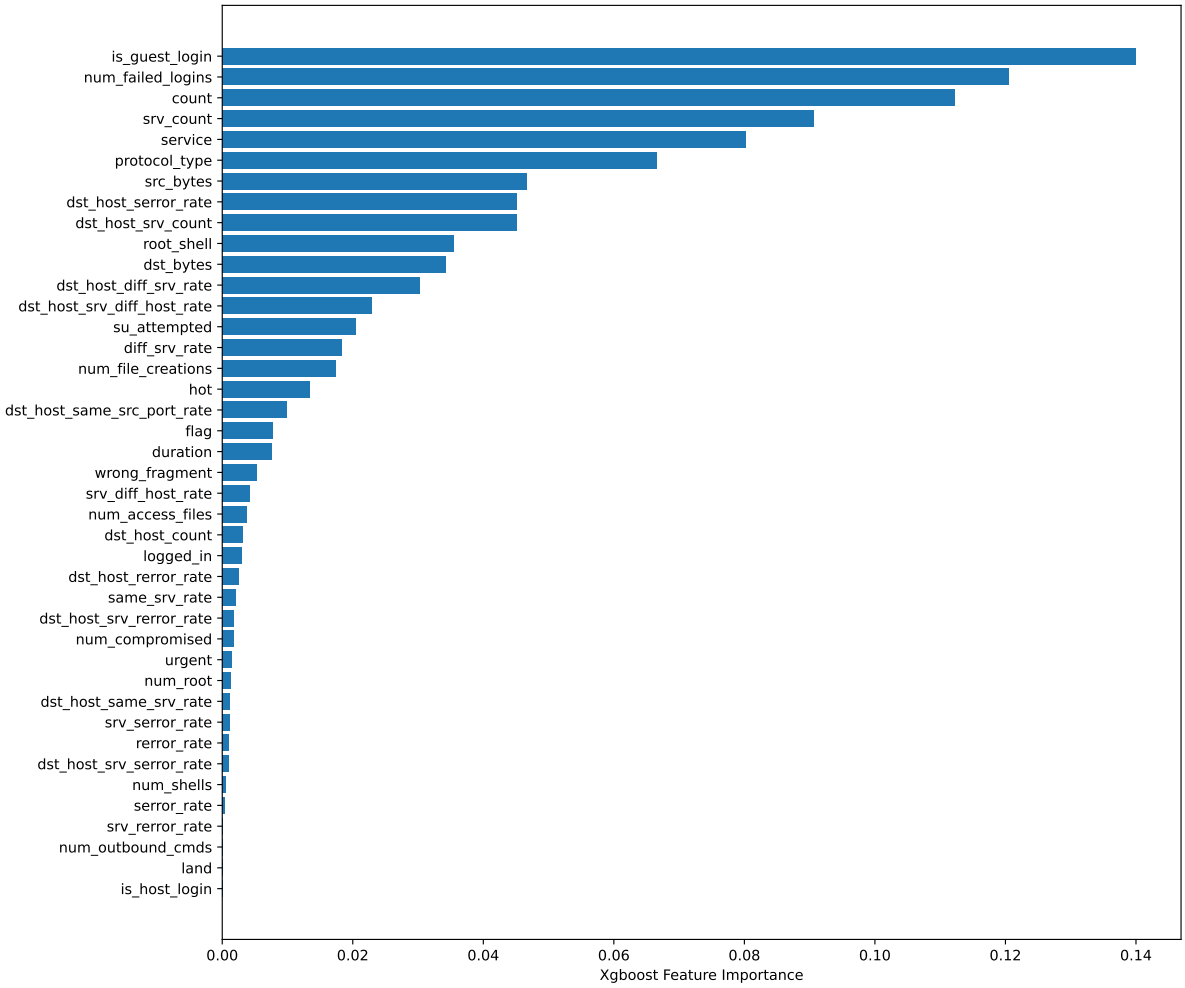


Figure 1: Feature importance graph by XGBoost algorithm.

To select the best features, we have conducted an extensive experiment using four different ML algorithms, including RF, DT, KNN, and MLP. Firstly, the output features are sorted in descending order, known as ranking features, for our feature selection process. After that, we calculate the accuracy of each of the algorithms using various subsets of features in the ranking features, which are the features that are generated from the XGBoost algorithm. Secondly, we calculate the accuracy of each of the algorithms using the k number of features where we set $k = N$ initially, where N is the number of features. If the accuracy results for all the algorithms satisfy an accuracy threshold Th_{acc}, this set of features will be nominated as a candidate features set. Then, we reduce the value of k by 2 and calculate the accuracy until $k > 0$. For instance, if we have a dataset of features size is 40 and if we take k=1, the time complexity of completing the feature selection process will be O(N) means it takes much more time with is equal to the number of features. If we take k=10, then its time complexity will be O(N/10) which is less complexity, but we can't easily find out the minimum number of features that can produce better accuracy as it produces a decrement of k by 10, which is 40, 30, 20, 10 features. So, we take k=2 so that we can get minimum time complexity which is O(N/2), and the minimum number of better feature sets of decrements of k by 2, which is 40, 38, 36, 34...N features without taking more or less than k=2. Moreover, this selection process helps us to clearly find out the variation of the accuracy rate while decreasing the number of features.

A set of features will be a candidate set if all the algorithms satisfy to achieve the accuracy greater or equal to the accuracy threshold Th_{acc}. Finally, from all the candidate sets of features, we choose the smallest set for which the accuracy of all algorithms can provide accuracy greater than Th_{acc}, which is 99.95%. The feature selection process is illustrated in the following graph 2. In the graph, we take the features and apply the XGBoost algorithm and initialize k=N, where N is the size of the features. Then we apply ML classifiers and check the accuracy rate of each classifier. If all the classifiers achieve the threshold hold accuracy rate, then those features are selected and stored as a candidate set of features; if not, then it will deduce the value of k by 2 to get the minimum number of features that will produce a better performance with the half of time complexity of total feature size (O(N/2)) and repeat the process.

The accuracy performance rate of different subsets of features is shown in Fig. 3. After analyzing all the accuracy results for all the ML algorithms for each subset, we find that after 20 feature selections, all ML algorithms satisfy the accuracy threshold value, which is 99.95%. So, we nominated these first top 20 features as a candidate feature set by which we produced in our proposed work.

3.2.3. Machine and Deep learning algorithms

In this work, we have used several ML and DL algorithms which are listed below, to perform binary and multilabel classification on our datasets.

- Random Forest (RF): An RF is a meta-approximation that employs averaging to enhance the accuracy level. Multiple decision tree classifiers are fitted to various sub-trials of the data set to prevent over-fitting [55]. It combines several uncut DTs derived from different bootstrap samples of the training data, and each attribute subset is sampled separately from the actual feature space [56]. Each tree and the class estimate a class that the majority of trees forecast becomes our model prediction [57]. It creates several independent decision trees from the training data set's original features and then votes to combine them into a single classification model

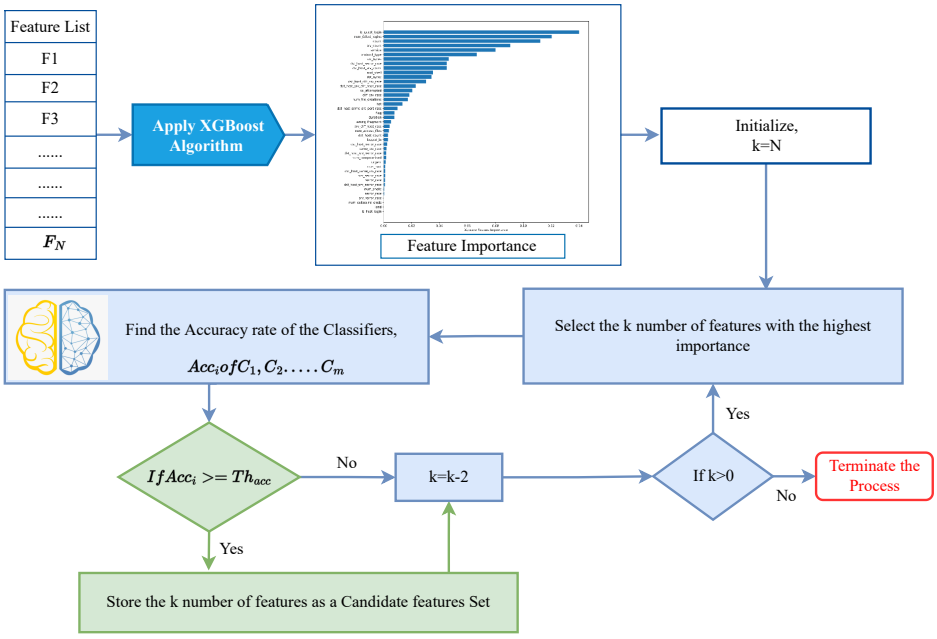


Figure 2: Feature selection process.

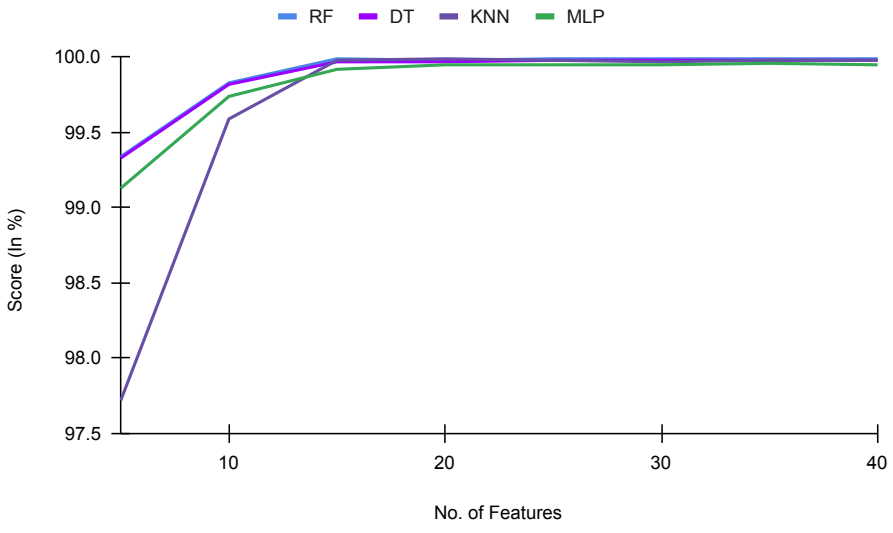


Figure 3: Feature importance results by XGBoost algorithm.

[58].

- Decision Tree (DT): Decision trees are a versatile tool that can be used in a variety of fields, including machine learning, image processing, and pattern recognition. DT is mainly used for grouping purposes. Furthermore, in Data Mining, DT is a commonly used classification model [59]. The root node, branches, and leaf nodes are the three main components of DT. The whole dataset is signified by the root node, which is alienated into two or more homogeneous sets, branches are the combination of features or attributes, and the output nodes are known as leaf nodes, where apartheid is halted [60]. Because of their ease of analysis and precision across a variety of data types, decision trees have a broad range of applications [61].

- K-Nearest Neighbour (KNN): KNN classifies unlabeled data instances by assigning them to another class of similar marked examples according to the closest measured distance between the data instance and the groups. Even lacking prior information about data distribution, KNN is simple to implement since it employs the Euclidean distance equation to calculate connections. It also generates precise end-user suggestions based on the simple application of similarity or distance for classifications [62, 63].

- Multilayer Perceptron (MLP): MLP is a common ANN architecture that consists of a series of layers made up of neurons and their connections. It can measure the weighted sum of its inputs before applying an activation function to generate a signal that will be sent to the next neuron [64]. It has one or more hidden layers between the input and output layers. The neurons are arranged in layers, connections are often guided from lower to upper layers, and the neurons in the same layer are not linked [65]. In the input layer, the number of neutrons equals the number of measurements for the pattern query, while the number of neurons in the output layer equals the number of classes [66].

- Convolution Neural Network (CNN): A CNN is a type of Artificial Neural Network (ANN) that is usually used to evaluate visual representations in deep learning [67]. Relying on the sharable structure of the convolution kernels or filters which move across input properties and give translational substance replies referred to as data maps, they're also defined as Shift Invariant or Space Invariant Artificial Neural Networks (SIANN) [68, 69]. The connecting arrangement among neurons matches the arrangement of the vertebrate neural activity, which was motivated by biological processes [70, 71, 72]. It is also a regularized variant of the multilayer perceptron [73], with a distinctive strategy for regularization: it uses the hierarchical structure in information and assembles motifs of evolving utilizing down into simpler patterns engraved in their filtering [74, 75].

- Artificial Neural Network (ANN): ANNs, sometimes known as neural networks (NNs), are computer systems modelled after the biological neurons seen in animal brains [76]. They are a set of interconnected systems or nodes in an ANN that roughly model neurons in a neocortex [77]. so each link, unlike synapses in the human brain, has the ability to send a response to neighboring neurons. An artificial neuron obtains the signals, analyses them, and afterwards sends signals to the neurons to which this is attached. Each neuron's signal is produced by certain non-linear functions of the combination of its inputs, and the "signal" at a link is a true number [76]. The interconnections are referred to as edges. The weight of synapses and edges varies as

training progresses. The weight affects the signal power at a connector. Synapses may well have a criterion at which a signal is just transmitted if indeed the cumulative activity exceeds it. Neurons are generally organized into layers. On their inputs, various layers may conduct distinct modifications. Signals go from the very first layer (the input layer)to the last layer (the output layer) [78, 79].

3.3. Proposed Architecture

Figure 4 depicts the schematic block diagram of the proposed approach. The approach consists of five sequential stages discussed as follows:

- Stage-1: In this stage, data preprocessing has been accomplished by handling missing values, scaling features, and encoding attribute values.
- Stage-2 In this stage, we check the dataset and apply SMOTE if the dataset is imbalanced to balance our datasets and handle the data imbalance problem.
- Stage-3: To exclude the less correlated features with the class label, the XGBoost algorithm is applied to obtain the most relevant features from the datasets.
- Stage-4: This stage splits the processed dataset to form training and testing dataset using K-fold cross-validation.
- Stage-5: In this phase, the algorithms are trained and tested to evaluate the performance in terms of several performance parameters, including accuracy, precision, recall, and f1-score. Afterwards, based on the performance, the highest-performed model is selected as a recommended model to detect network intrusion, and then the model is compared with other existing models.

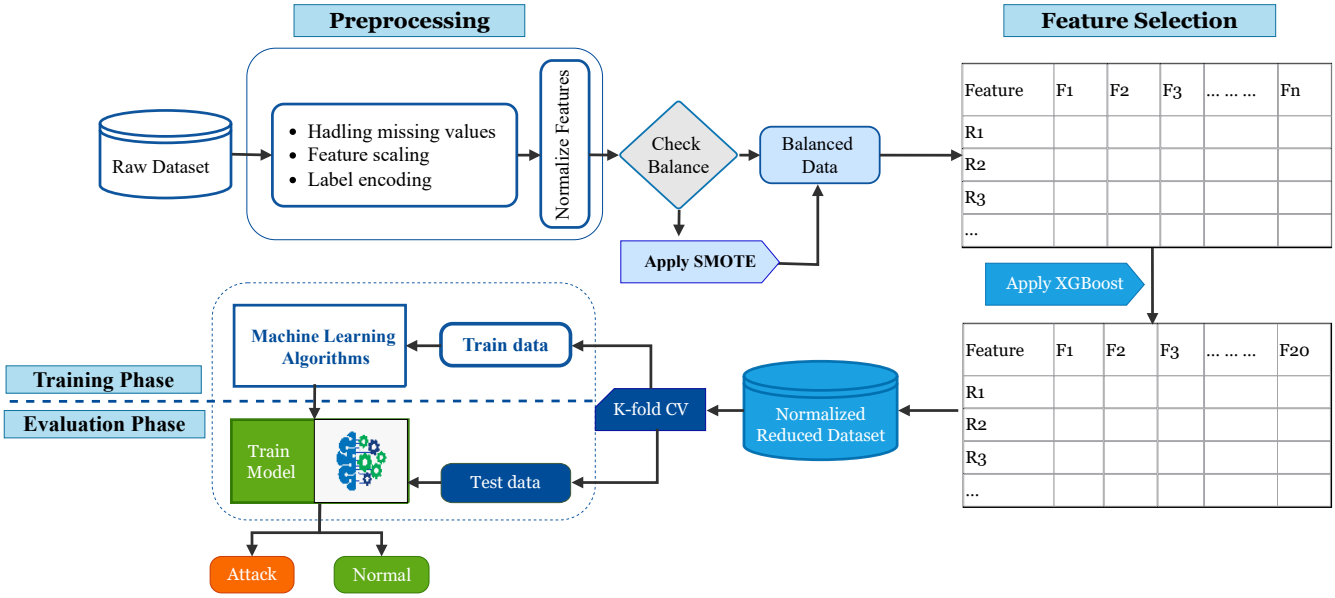


Figure 4: Proposed framework for feature selection and intrusion detection.

Category	Description	Sub-category	Count
dos	Turn off a network rendering it unreachable to its target purposes	back, land, Neptune, pod, smurf, and teardrop	391458
u2r	Illegal access from a distant computer	buffer_overflow, loadmodule, Perl and rootkit	52
r2l	Illegal access to superuser (root) credentials on the local machine	ftp_write, guess_passwd, imap, multihop, phf, spy, aware client, and ware master	1126
probe	Sends packets to a remote machine to acquire local access.	ipsweep, Nmap, port sweep, and satan	4107
normal	Normal packets	normal	97278

Table 3: Sub-categories in each class.

4. Model Implementation and Evaluation

We have implemented a novel hybrid approach to detect network intrusion by utilizing SMOTE for data balancing and XGBoost for feature selection and applying several ML and DL algorithms to analyze and select the best model. The approach is validated and assessed using extensive experiments on various datasets. In the following section, we described the dataset descriptions followed by the data preparation and training process.

4.1. Dataset Description

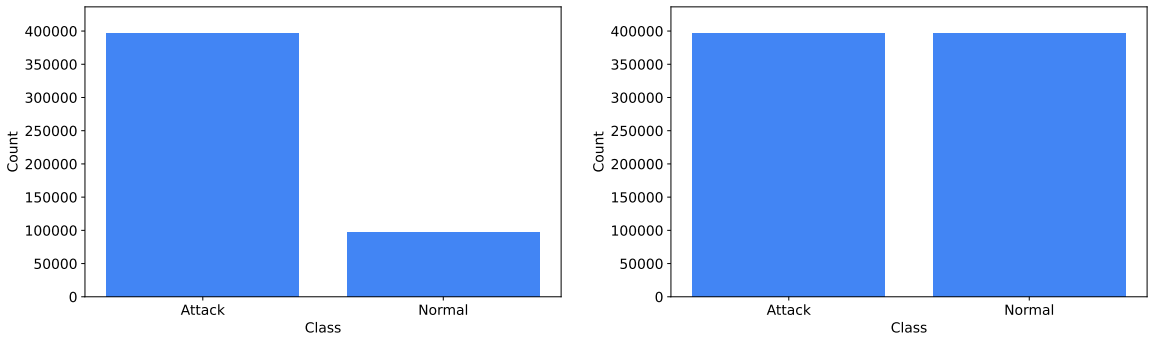
This paper tests several ML and DL algorithms on the KDDCUP'99 and CIC-MalMem-2022 (MalMemAnalysis-2022) datasets.

4.1.1. KDDCUP'99 Dataset

Knowledge Discovery and Data Mining (KDD)Cup 1999 dataset, [80] which is a benchmark dataset released by the US Department of Defense Advanced Research Projects Agency (DARPA). The dataset was created to forecast a model differentiating between bad connections, known as intrusions or attacks, and regular connections. The dataset comprises a standard set of auditable data covering a wide range of virtual intrusions in a military network environment. A communication link is defined as a series of TCP packets sent from the source IP address to the target IP address over a fixed period with data being transmitted from the source IP address to the target IP address according to a pre-configured protocol [81]. This is the most broadly used dataset for network intrusion detection assessment [82], and it has been available for a long time. In our experiment, we used 10% of the KDDCUP'99 dataset ('kddcup.data_10_percent.gz') to reduce the experimental time and cost. It has 41 features(input), 23 subcategories (outcome), 5 categories (attack_type) and 2 labels of attacks (label) with 44 attributes. Table 3 represents the different sub-categories of the various attacks, while Figure 5 and 6depicts the distribution of attack categories before and after SMOTE using a bar chart both for binary and multilabel classification. All the features are described in the following Table 4.

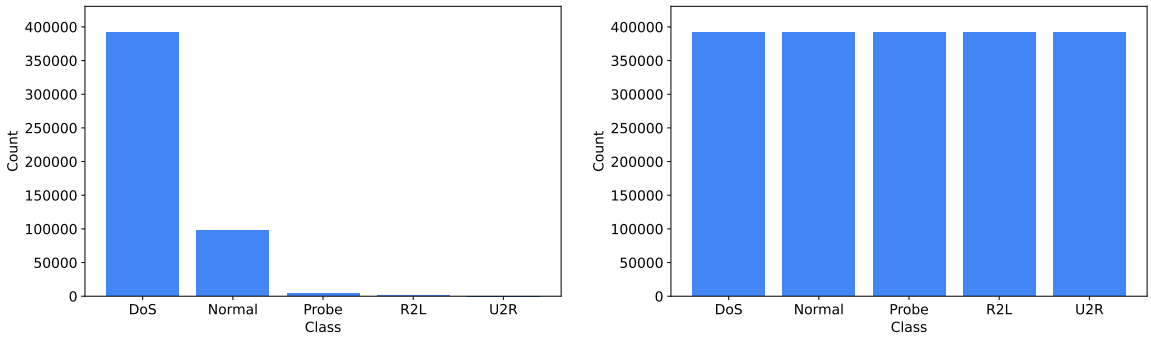
4.1.2. CIC-MalMem-2022

Malware that conceals to escape detection and elimination is known as obfuscated malware. The Obfuscated dataset [35] is a memory-based evaluation of obfuscated malware identification algorithms. It is a Malware Memory Analysis-2022 dataset. The data was intended to mimic a real scenario as closely as feasible by utilizing malware widely used in the real world. It's a fair dataset composed of Spyware, Ransomware, including Trojan Horse malware which may be employed to evaluate obfuscated malware monitoring systems. The dataset is balanced, with half of it conjured up of malignant memory dumps while half of it is composed of benign memory dumps. There are 58,596



(a) Before SMOTE (b) After SMOTE

Figure 5: Before and After the distribution of attacks in KDDCUP'99 for binary classification.



(a) Before SMOTE (b) After SMOTE

Figure 6: Before and After the distribution of attacks in KDDCUP'99 for multilabel classification.

SI.No.	Feature	Type	SI.No	Feature	Type
0	Duration	int64	22	Count	int64
1	protocol_type	object	23	srv_count	int64
2	Service	object	24	serror_rate	float64
3	Flag	object	25	srv_serror_rate	float64
4	src_bytes	int64	26	rerror_rate	float64
5	dst_bytes	int64	27	srv_rerror_rate	float64
6	Land	int64	28	same_srv_rate	float64
7	wrong_fragment	int64	29	diff_srv_rate	float64
8	Urgent	int64	30	srv_diff_host_rate	float64
9	Hot	int64	31	dst_host_count	int64
10	num_failed_logins	int64	32	dst_host_srv_count	int64
11	logged_in	int64	33	dst_host_same_srv_rate	float64
12	num_compromised	int64	34	dst_host_diff_srv_rate	float64
13	root_shell	int64	35	dst_host_same_src_port_rate	float64
14	su_attempted	int64	36	dst_host_srv_diff_host_rate	float64
15	num_root	int64	37	dst_host_serror_rate	float64
16	num_file_creations	int64	38	dst_host_srv_serror_rate	float64
17	num_shells	int64	39	dst_host_rerror_rate	float64
18	num_access_files	int64	40	dst_host_srv_rerror_rate	float64
19	num_outbound_cmds	int64	41	Outcome	object
20	is_host_login	int64	42	attack_type	object
21	is_guest_login	int64	43	label	object

Table 4: Features in the KDD-CUP'99 dataset.

entries in total, comprising 29,298 benign as well as 29,298 harmful entries. Figure 7 illustrates the distribution of attack categories using a bar chart for binary classification. All the features are described in the following Table 5.

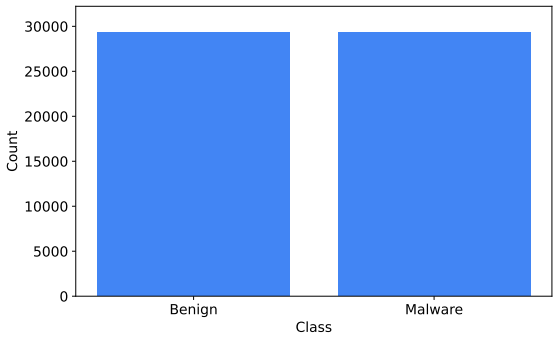


Figure 7: Distribution of attacks in CIC-MalMem-2022

4.2. Data Preparation

The preprocessed dataset is fed into the ML and DL models. Preprocessing is accomplished by handling missing values, scaling features, and selecting a particular number of features.

4.2.1. Handling Missing values

The missing value occurs in the dataset due to data corruption or not recording data properly. Handling missing values is crucial in the data preprocessing phase, as many ML and DL algorithms trained with such data output incorrect results. We adopt simple methods to handle the missing value of the attributes. For example, missing values are handled by removing rows containing nan (null value) in our dataset, -inf, inf, and duplicate entries.

4.2.2. Feature Scaling using Standardization

Feature scaling refers to a procedure for normalizing the value of features within a certain range. We have employed a standardization method to scale the value of features in the dataset. Standardizing the features' value increases the accuracy of the model. In many use cases, the model's accuracy depends on the nature of the dataset utilized for training the model. The accuracy of ML and DL models is significantly reduced if the features' values have different measurement units or imbalanced range differences. To handle this issue, the standardization method is used to make attributes' values within a certain balanced range. Each value of an attribute is normalized by subtracting the mean and dividing by the standard deviation of that attribute [83]. Features' values are normalized using the equation6.

$$X_{st} = \frac{x - mean(x)}{std(x)} \tag{6}$$

Here, X_{st} is the standardized value, x is the actual value of an attribute, $mean(x)$ is the mean of actual value and $std(x)$ is the standard deviation of actual value.

SI.No	Feature	Type	SI.No	Feature	Type
0	Category	object	29	malfind .protection	int64
1	pslist.nproc	int64	30	malfind.uniqueInjections	float64
2	pslist.nppid	int64	31	psxview.not_in_pslist	int64
3	pslist.avg_threads	float64	32	psxview.not_in_eprocess_pool	int64
4	pslist.nprocs64bit	int64	33	psxview.not_in_ethread_pool	int64
5	pslist.avg_handlers	float64	34	psxview.not_in_pspcid_list	int64
6	dlllist.ndlls	int64	35	psxview.not_in_csrss_handles	int64
7	dlllist.avg_dlls_per_proc	float64	36	psxview.not_in_session	int64
8	handles.nhandles	int64	37	psxview.not_in_deskthrd	int64
9	handles.avg_handles_per_proc	float64	38	psxview.not_in_pslist_false_avg	float64
10	handles.nport	int64	39	psxview.not_in_eprocess_pool_false_avg	float64
11	handles.nfile	int64	40	psxview.not_in_ethread_pool_false_avg	float64
12	handles.nevent	int64	41	psxview.not_in_pspcid_list_false_avg	float64
13	handles.ndesktop	int64	42	psxview.not_in_csrss_handles_false_avg	float64
14	handles.nkey	int64	43	psxview.not_in_session_false_avg	float64
15	handles.nthread	int64	44	psxview.not_in_deskthrd_false_avg	float64
16	handles.ndirectory	int64	45	modules.nmodules	int64
17	handles.nsemaphore	int64	46	svcscan.nservices	int64
18	handles.ntimer	int64	47	svcscan.kernel_drivers	int64
19	handles.nsection	int64	48	svcscan.fs_drivers	int64
20	handles.nmutant	int64	49	svcscan.process_services	int64
21	ldrmodules.not_in_load	int64	50	svcscan.shared_process_services	int64
22	ldrmodules.not_in_init	int64	51	svcscan.interactive_process_services	int64
23	ldrmodules.not_in_mem	int64	52	svcscan.nactive	int64
24	ldrmodules.not_in_load_avg	float64	53	callbacks.ncallbacks	int64
25	ldrmodules.not_in_init_avg	float64	54	callbacks.nanonymous	int64
26	ldrmodules.not_in_mem_avg	float64	55	callbacks.ngeneric	int64
27	malfind.ninjections	int64	56	Class	object
28	malfind.commitCharge	int64			

Table 5: Features for CIC-MalMem-2022.

4.2.3. Label Encoding

Label encoding is the procedure of converting category data to numerical values. To develop a model in an ML method, we must transform or encode categorical features into quantitative data in order to enter the data into the training module and develop a model. By replacing the categorical values with the value 0 to the total number of classes (n)-1 [66]. We can use 0, 1, 2, 3, and 4 in place of all these various classes if the categorical data contain 5 different classes. The label encoding technique for binary and multilabel for the KDDCUP'99 dataset is shown in Tables 6 and 7.

Attack types	Label encoding
Attack	0
Normal	1

Table 6: Label encoding process.

Attack types	Label encoding
DoS	0
Normal	1
Probe	2
R2L	3
U2R	4

Table 7: Label encoding process.

4.3. Training Process

The training process has been done using HP 250 G5 Notebook PC with Microsoft Windows 10 Pro 64bit (10.0.19042 build 19042) OS, Intel(R) Core (TM) i3-6006U CPU @ 2.00GHz with 8GB RAM. We have used the Jupyter Notebook 6.4.6 tool, and for the programming language, Python 3.8.5 has been used to implement the models. Pandas 1.3.4 and NumPy 1.19.5 frameworks have been used for data cleaning, extraction, feature selection and Matplotlib 3.5.0, Seaborn 0.11.2 framework for data visualization, and finally, Scikit-learn 0.24.1 package for data analysis. To evaluate the performance of our proposed method, we have used some metrics, including accuracy, precision, recall, f1-score, RMSE, and ROC Curve.

5. Result Analysis

The extension results are analyzed on various performance metrics to find the best model to detect network intrusion. We analyzed the performance based on all the features, selected features, and proposed features. The finding showed that the performance of our proposed feature outperforms the other two features.

We have conducted experiments for multiclass and binary class-based intrusion detection. We have used k-fold cross-validation with the value of k=10 shown in Figure 8. While 80% of the dataset's data have been selected for

ML	Accuracy			Precision			Recall			F1-score			RMSE		
	All	Selected	Proposed	All	Selected	Proposed	All	Selected	Proposed	All	Selected	Proposed	All	Selected	Proposed
RF	99.99	99.99	99.99	99.97	99.97	99.99	99.99	99.99	99.99	99.98	99.98	99.99	1.19	1.19	1.18
DT	99.98	99.97	99.98	99.96	99.94	99.98	99.97	99.95	99.98	99.96	99.95	99.98	1.56	1.86	1.46
KNN	99.97	99.97	99.98	99.94	99.95	99.98	99.95	99.95	99.98	99.95	99.95	99.98	1.86	1.74	1.33
MLP	99.91	99.93	99.95	99.87	99.87	99.95	99.86	99.9	99.95	99.87	99.89	99.95	2.92	2.7	2.22

Table 8: Performance results for binary classification.

training, and the remaining 20% of data are being tested, dividing the whole data into 10 different folds and taking a different fold every time during the experiments.

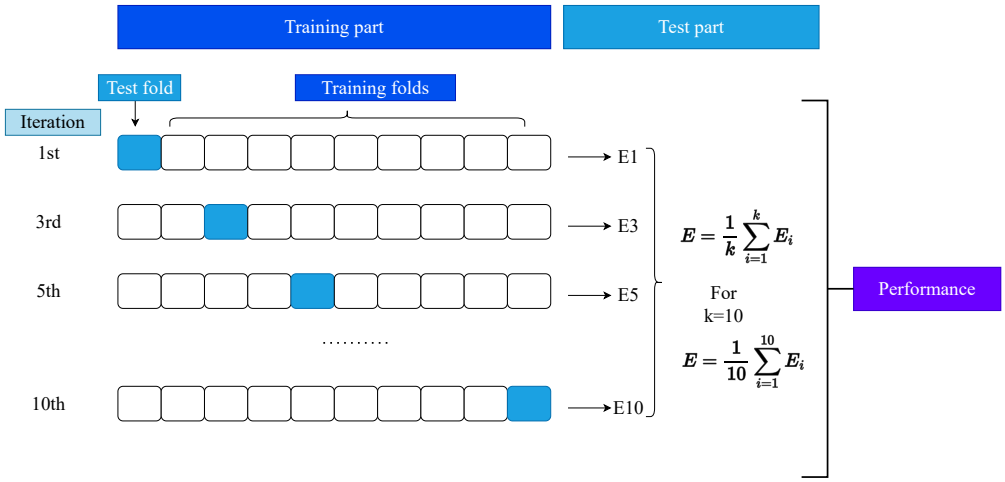


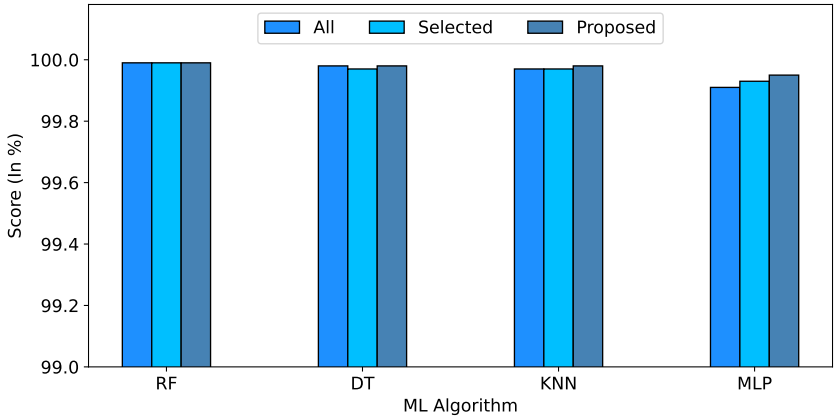
Figure 8: K-fold cross-validation.

5.1. Performance Analysis of KDDCUP'99 Dataset

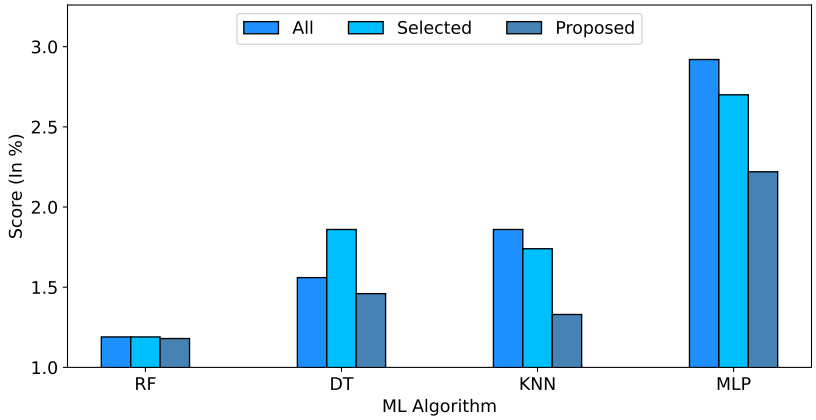
The performance comparison results for binary classification for the KDDCUP'99 dataset are illustrated in Table 8 and Fig. 9 in tabular and bar chart format, respectively. The experiment results for considering all features, 20 features without applying smote, and the proposed model's performance is represented. Here, all indicate that the features that are not selected and not applied smote are just pre-proposed, scaled, and then applied to the machine learning algorithms to build models and evaluate the performance. On the other hand, the proposed model indicates all the proposed methodology processes by which evaluation has been measured.

The accuracy rate of our proposed scheme in binary classification is substantially higher, as shown in the Table and Bar Chart. The accuracy rising rate among the three bar graphs is high, and the RMSE rate is significantly lower than all selected features.

From the binary comparison graph fig 9, the accuracy of the proposed model for RF, DT, KNN, and MLP is 99.99%, 99.98%, 99.98%, and 99.95%, respectively. From the graph fig 9(a), it is clear that the accuracy increases by considering all the features with the proposed model are 0%, 0%, 0.01%, 0.04%; by considering selected features with the proposed model are 0%, 0.01%, 0.01%, 0.02% for RF, DT, KNN, MLP respectively that proves the effectiveness of the proposed models with a smaller number of features. Fig 9(b) shows a variation in the RMSE values for different features and the proposed model. By considering all the features with the proposed models the reduced



(a) Accuracy



(b) RMSE

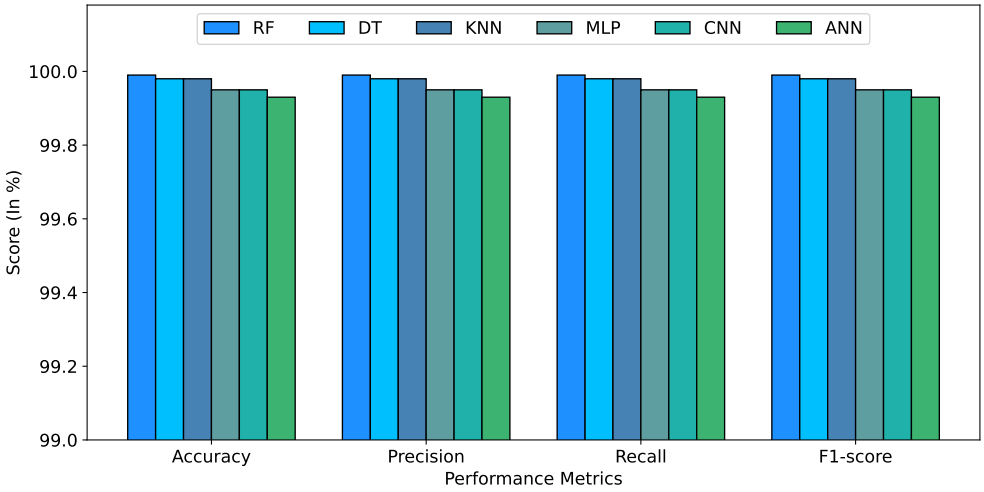
Figure 9: Performance comparison graphs for binary classification.

ML	Accuracy	Precision	Recall	F1-score	MAE	MSE	RMSE
CNN	99.95	99.95	99.95	99.95	0.05	0.05	2.27
ANN	99.93	99.93	99.93	99.93	0.07	0.07	2.68

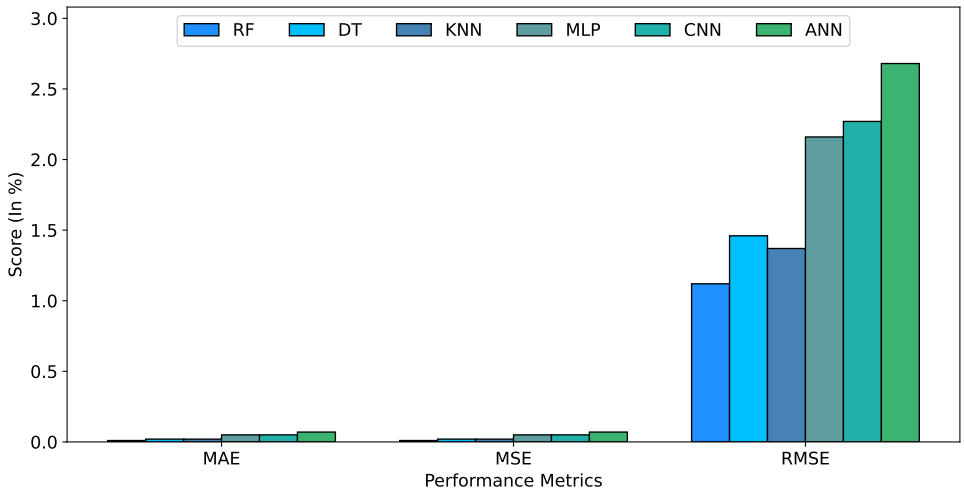
Table 9: Performance analysis for binary classification.

RMSE error are 0.01%, 0.1%, 0.53%, 0.70%; considering selected features with the proposed models the reduced RMSE error are 0.01%, 0.40%, 0.41%, 0.48% for RF, DT, KNN, MLP respectively.

We also explore two neural network models, namely ANN and CNN, in our approach, where in Table 9 shows that the performance of CNN is greater than ANN, which is 0.02%.



(a) Performance

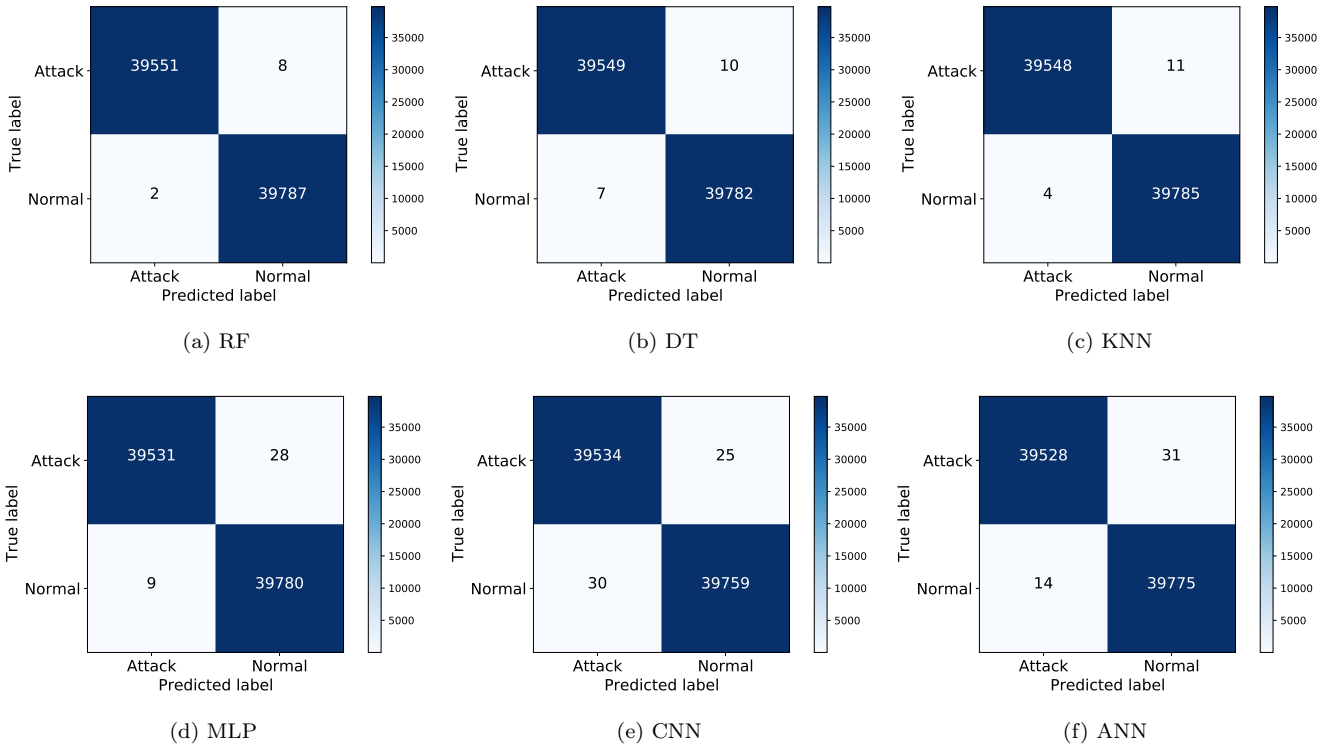


(b) Error

Figure 10: Performance analysis graphs for binary classification.

The Performance analysis graphs for binary classification are shown in fig 10 where the accuracy rate among all the algorithms RF gives the better performance, which is 99.99%, and ann gives lower accuracy, which is 99.93%.

The RMSE error rate for RF is 1.18%, and ANN is 2.68%.



(a) RF (b) DT (c) KNN

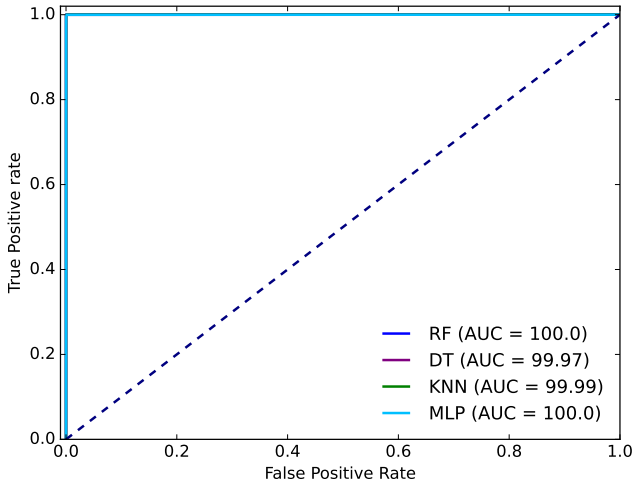
(d) MLP (e) CNN (f) ANN

Figure 11: Confusion matrix for binary classification of KDDCUP'99.

Fig 11 shows the confusion matrix for all the proposed algorithms. The TP, TN, FP, FN rates are 49.84%, 50.14%, 0%, 0.01%; 49.84%, 50.14%, 0.01%, 0.01%; 49.84%, 50.14%, 0.01%, 0.01%; 49.82%, 50.13%, 0.01%, 0.04%; 49.82%, 50.12%, 0.02%, 0.03%; 49.8%, 50.13%, 0.01%, 0.06% for RF, DT, KNN, MLP, CNN and ANN respectively. From the confusion matrix results, it is visible that RF outperforms the other algorithms using 20 features with higher rates of TP and TN with very lower rates of FP and FN. A random forest is essentially a set of individual decision trees that work together to form an ensemble. Thus, multiple learners' algorithms can be trained with the knowledge to achieve the highest accuracy.

Fig 12 shows the Binary ROC Curve for KDDCUP'99, where the AUC score for RF, DT, KNN, and MLP are 100%, 99.97%, 99.99%, and 100%, respectively. Among all the RF and MLP give the highest accuracy rate. MLP consists of a series of layers made up of neurons and their connections. It has one or more hidden layers between the input and output layers where the neurons are arranged in layers and connections are often guided from lower to upper layers.

The performance comparison results for multilabel classification for the KDDCUP'99 dataset are illustrated in Table 10 and Fig. 13 in tabular and bar chart format, respectively. The experiment results for considering all features, 20 features without applying smote, and the proposed model's performance is represented. Here, all indicate that the features that are not selected and not applied smote are just pre-proposed, scaled, and then applied to the machine learning algorithms to build models and evaluate the performance. On the other hand, the proposed model indicates all the proposed methodology processes by which evaluation has been measured.



(a) ROC Curve

Figure 12: Binary ROC Curve for KDDCUP'99.

ML	Accuracy			Precision			Recall			F1-score			RMSE		
	All	Selected	Proposed	All	Selected	Proposed	All	Selected	Proposed	All	Selected	Proposed	All	Selected	Proposed
RF	99.99	99.98	99.99	99.99	99.94	99.99	92.94	92.9	99.99	95.8	95.75	99.99	2.06	2.42	0.9
DT	99.97	99.96	99.99	99.72	92.67	99.99	92.55	92.54	99.99	95.47	92.6	99.99	2.92	3.31	1.9
KNN	99.96	99.97	99.98	99.42	99.72	99.98	92.63	92.59	99.98	95.36	95.48	99.98	3.18	2.81	2.4
MLP	99.93	99.92	99.93	78.75	77.91	99.93	78.6	78.84	99.93	78.68	78.36	99.93	4.52	4.82	4.75

Table 10: Performance results for multilabel classification.

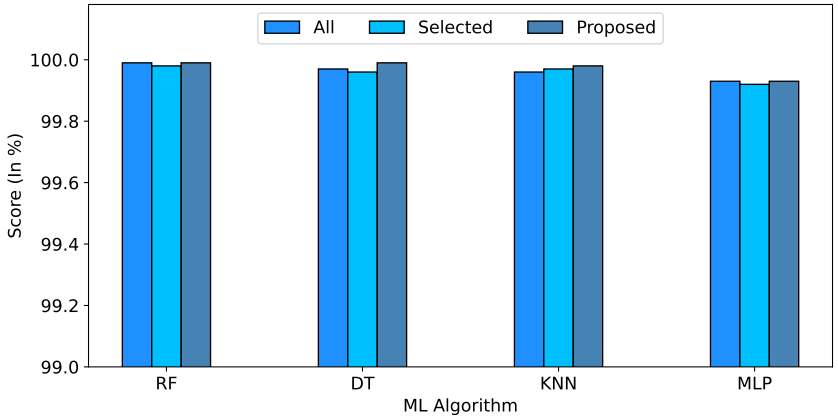
The accuracy rate of our proposed scheme in multilabel classification is substantially higher, as shown in the bar chart. The accuracy rising rate among the three bar graphs is high, and the RMSE rate is significantly lower than all selected features.

From the multilabel comparison graph fig 13, the accuracy of the proposed model for RF, DT, KNN, and MLP is 99.99%, 99.99%, 99.98%, and 99.93%, respectively. From graph 13(a), it is clear that the accuracy increases by considering all the features with the proposed model are 0%, 0.02%, 0.02%, 0%; by considering selected features with the proposed models are 0.01%, 0.03%, 0.01%, 0.01% for RF, DT, KNN, MLP respectively that proves the effectiveness of the proposed model. 13(b) shows a variation in the RMSE values for different features and the proposed models. By considering all the features with the proposed models the reduced RMSE error is 1.16%, 1.02%, 0.78% and 0.23%; considering selected features with the proposed models the reduced RMSE error is 1.52%, 1.41%, 0.41%, 0.07% for RF, DT, KNN, MLP respectively.

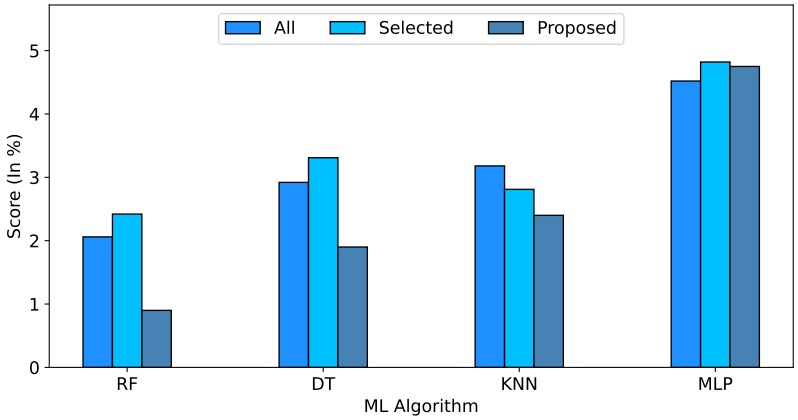
We also explore two neural network models, namely ANN and CNN, in our approach, wherein Table 11 shows

ML	Accuracy	Precision	Recall	F1-score	MAE	MSE	RMSE
CNN	99.84	99.84	99.84	99.84	0.26	0.49	6.97
ANN	99.86	99.86	99.86	99.86	0.22	0.41	6.42

Table 11: Performance analysis for multilabel classification.



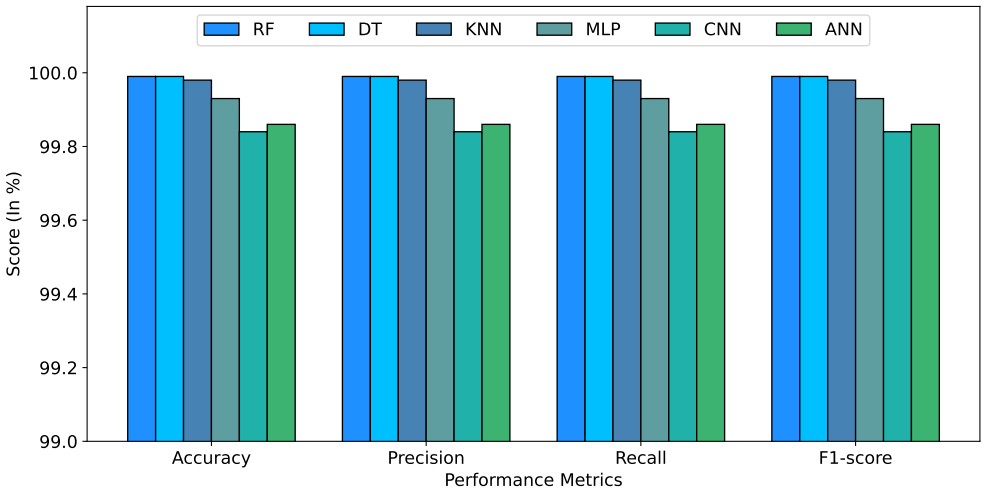
(a) Accuracy



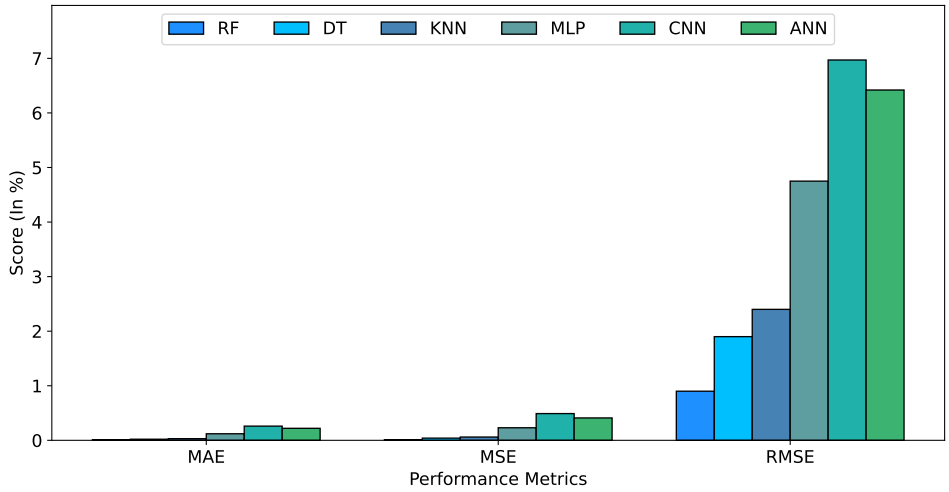
(b) RMSE

Figure 13: Performance comparison graphs for multilabel classification.

that the performance of ANN is greater than CNN, which is 0.02%.



(a) Performance

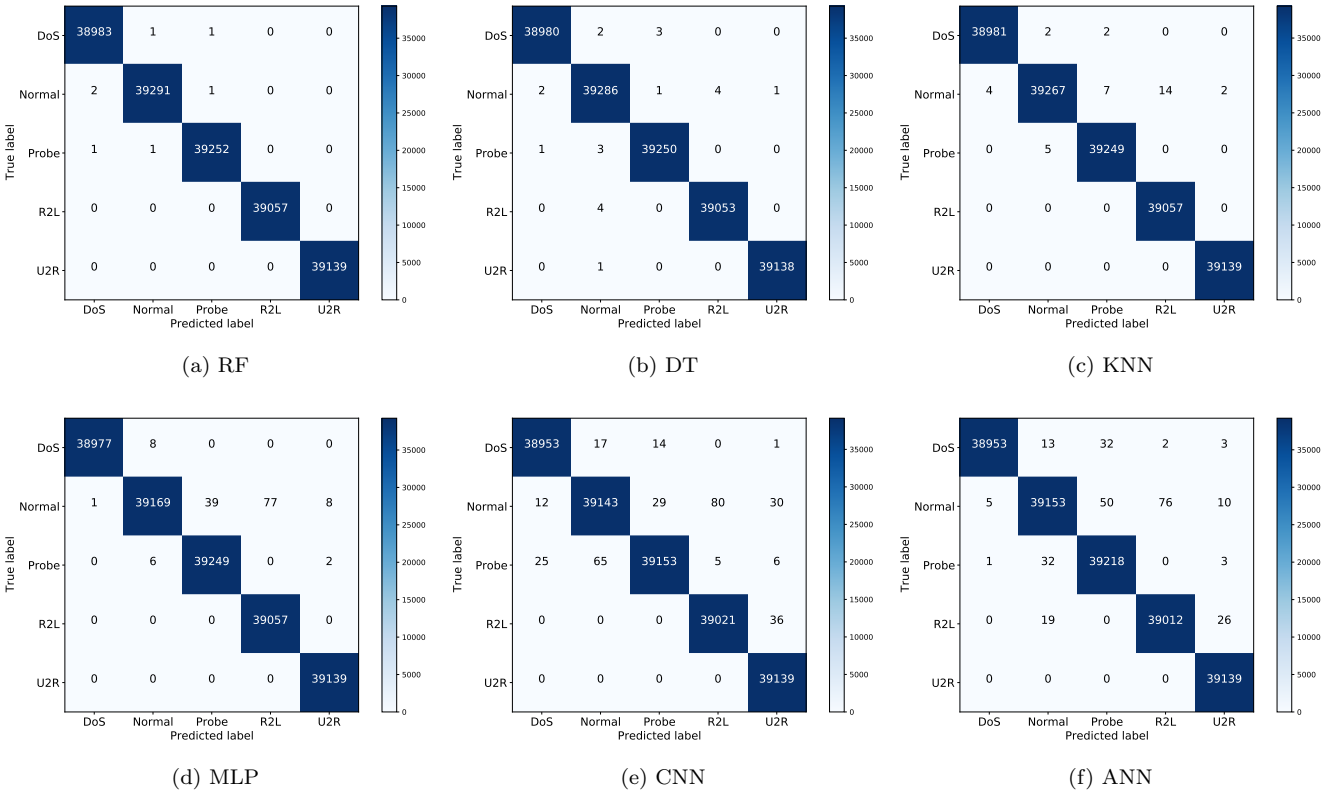


(b) Error

Figure 14: Performance analysis graphs for multilabel classification.

The Performance analysis graphs for multilabel classification are shown in fig 14 where the accuracy rate among all the algorithms RF and DT gives better performance, which is 99.99%, and CNN gives lower accuracy, which is 99.84%. The RMSE error rate for RF is 0.9%, and ANN is 6.97%.

Fig 15 shows the confusion matrix for all the proposed algorithms. Among all the confusion matrices, RF and DT produce a large number of TP, TN, and a very smaller number of FP, and FN rates, which gives a better performance of detecting intrusion detection with proper accuracy rate. For RF, the TP, TN, FP, FN rate are 19.92%, 80.08%, 0%, 0% ; 20.07%, 79.92%, 0%, 0% ; 20.05%, 79.94%, 0%, 0% ; 19.95%, 80.04%, 0%, 0% ; 20%, 80%, 0%, 0% ; considering DoS, Normal, Probe, R2L, U2R respectively. The TP, TN, FP, FN rate are 19.92%, 80.08%, 0%, 0% ; 20.07%, 79.92%, 0%, 0.01% ; 20.05%, 79.94%, 0%, 0% ; 19.95%, 80.04%, 0%, 0% ; 19.99%, 80%, 0%, 0% ; considering DoS, Normal, Probe, R2L, U2R respectively for DT. Random Forest produces superior findings, operates well on huge datasets, and can create estimates for missing data. It is a collection of independent

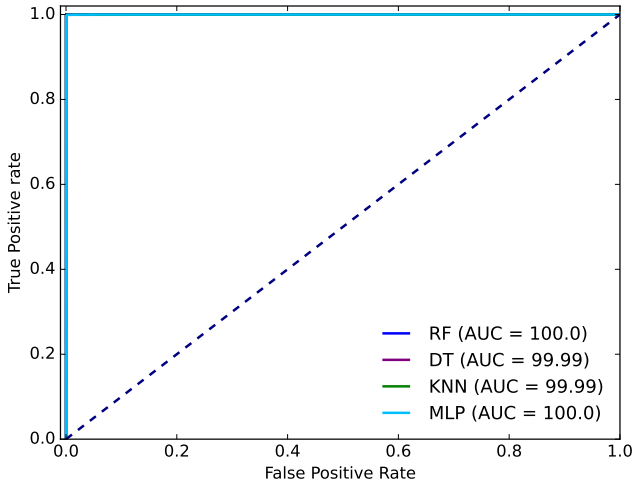


(a) RF (b) DT (c) KNN

(d) MLP (e) CNN (f) ANN

Figure 15: Confusion matrix for multilabel classification of KDDCUP'99.

decision trees collaborating to form an ensemble. As a result, knowledge can be used to train several learner algorithms to reach maximum accuracy. On the other hand, DT is simple to use, comprehend and explain; it takes minimal time and effort to plan and produce; it may be used in conditional probability-based reasoning and can offer strategic responses to uncertain situations.



(a) ROC Curve

Figure 16: Multilabel ROC Curve for KDDCUP'99.

ML	Accuracy		Precision		Recall		F1-score		MAE		MSE		RMSE	
	All	Proposed	All	Proposed	All	Proposed	All	Proposed	All	Proposed	All	Proposed	All	Proposed
RF	100	100	100	100	100	100	100	100	0	0	0	0	0	0
DT	100	100	100	100	100	100	100	100	0	0	0	0	0	0
KNN	99.97	99.97	99.97	99.97	99.97	99.97	99.97	99.97	0.03	0.03	0.03	0.03	1.85	1.85
MLP	100	100	100	100	100	100	100	100	0	0	0	0	0	0
CNN	99.98	99.98	99.98	99.98	99.98	99.98	99.98	99.98	0.02	0.02	0.02	0.02	1.31	1.31
ANN	100	100	100	100	100	100	100	100	0	0	0	0	0	0

Table 12: Performance comparison analysis for binary classification.

Fig 16 shows the multilabel ROC Curve for KDDCUP'99, where the AUC score for RF, DT, KNN, and MLP are 100%, 99.97%, 99.99%, and 100%, respectively. Among all the RF and MLP give the highest accuracy rate. MLP consists of a series of layers made up of neurons and their connections. It has one or more hidden layers between the input and output layers where the neurons are arranged in layers, and connections are often guided from lower to upper layers.

After analyzing all the ML and DL algorithms, we can find that RF outperforms other ML and DL algorithms both for binary and multilabel classifications to detect network intrusion.

5.2. Performance Analysis of CIC-MalMem-2022 Dataset

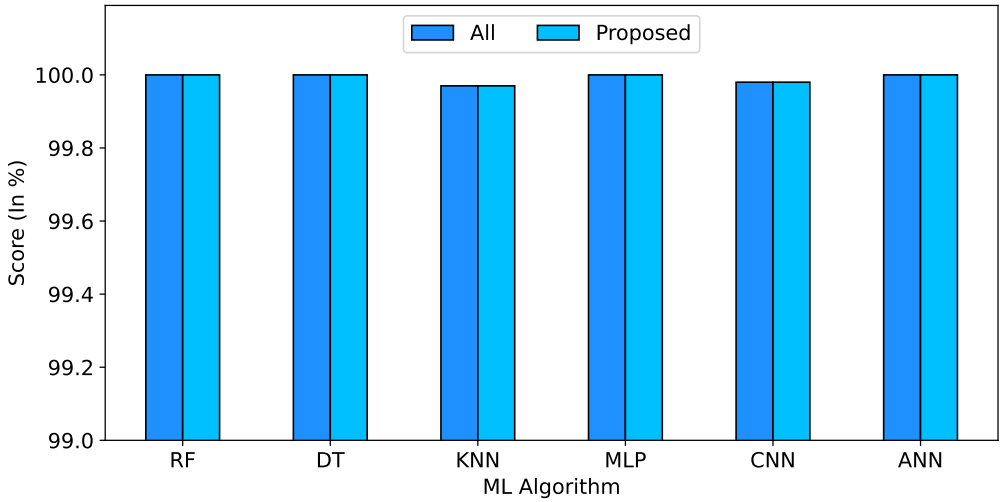
The performance comparison results for binary classification for the CIC-MalMem-2022 dataset are illustrated in Table 12 and Fig. 17 in tabular and bar chart format, respectively. The experiment results for considering all features and the proposed model's performance are represented. Here, all indicate that the features are just pre-proposed, scaled, and then applied to the machine learning algorithms to build models and evaluate the performance. On the other hand, the proposed model indicates all the proposed methodology processes by which evaluation has been measured.

The accuracy rate of our proposed scheme in binary classification is substantially the same as all features, as shown in the Table and Bar Chart. The accuracy rate between the two bar graphs is almost the same, and the RMSE rate is also approximately the same as all features.

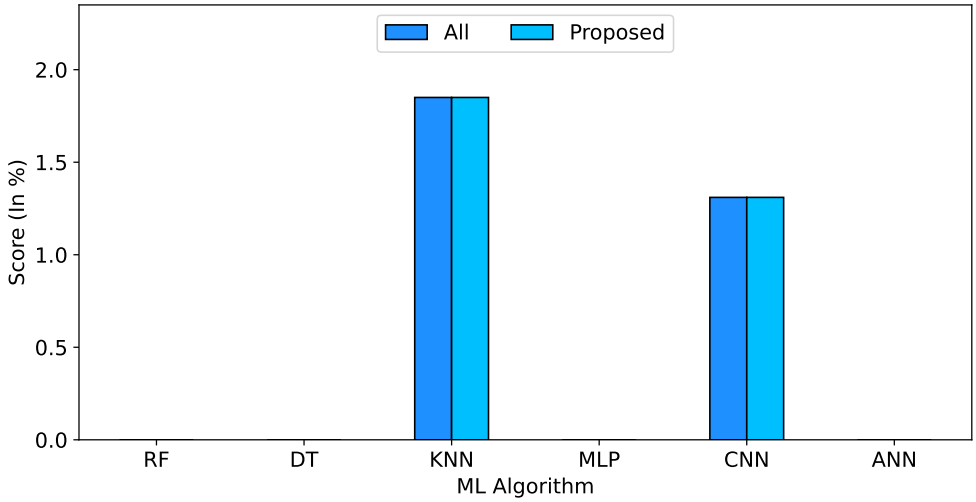
From the binary comparison graph fig 17, the accuracy of the proposed model for RF, DT, KNN, MLP, CNN, and ANN is 100%, 100%, 99.97%, 100%, 99.98%, and 100%, respectively. From the graph fig 17(a), it is clear that the accuracy increasing rate by considering all the features with the proposed model is the same for all ML algorithms that prove the effectiveness of the proposed model with a smaller number of features. Fig 17(b) shows the same RMSE values for all features and the proposed features.

The Performance analysis graphs for binary classification are shown in fig 18 where, the accuracy rate among all the algorithms RF, DT, MLP, and ANN gives the better performance, which is 100%, and KNN gives lower accuracy, which is 99.97%. The RMSE error rate for RF,DT,MLP,ANN is 0% and KNN is 1.85%.

Fig 19 shows the confusion matrix for all the proposed algorithms. The TP, TN, FP, FN rates are 51.41%, 48.59%, 0%, 0% ; 51.41%, 48.59%, 0%, 0% ; 51.39%, 48.57%, 0.02%, 0.02% 51.41%, 48.59%, 0%, 0% ; 51.41%, 48.57%, 0.02%, 0% ; 51.41%, 48.59%, 0%, 0% ; for RF, DT, KNN, MLP, CNN and ANN respectively. The confusion

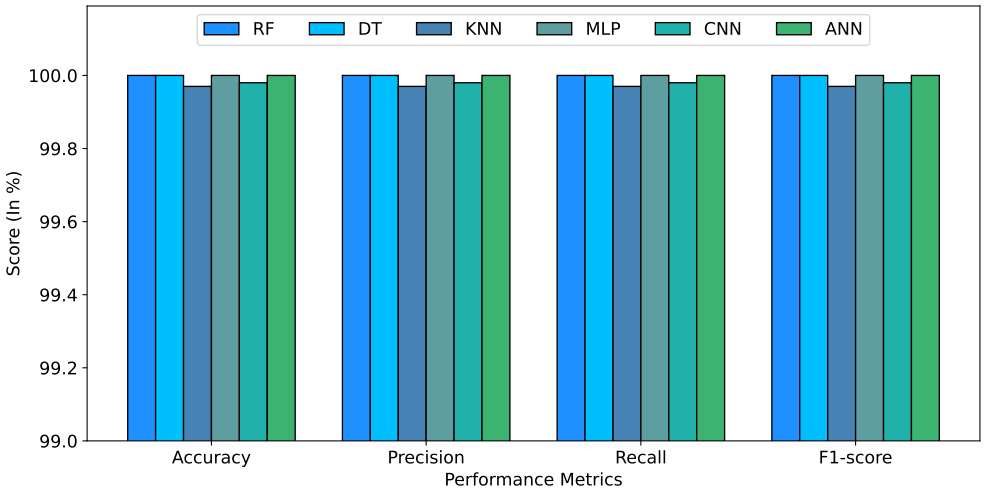


(a) Accuracy

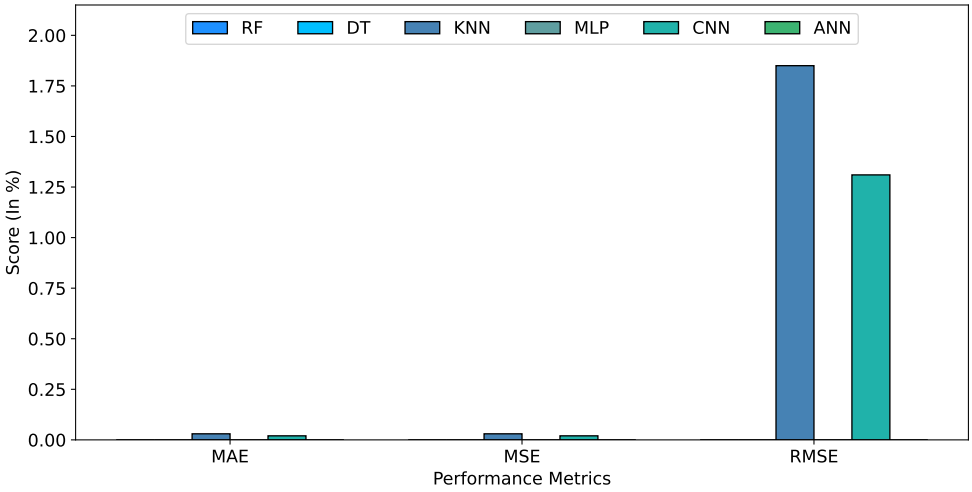


(b) RMSE

Figure 17: Performance comparison graphs for binary classification.

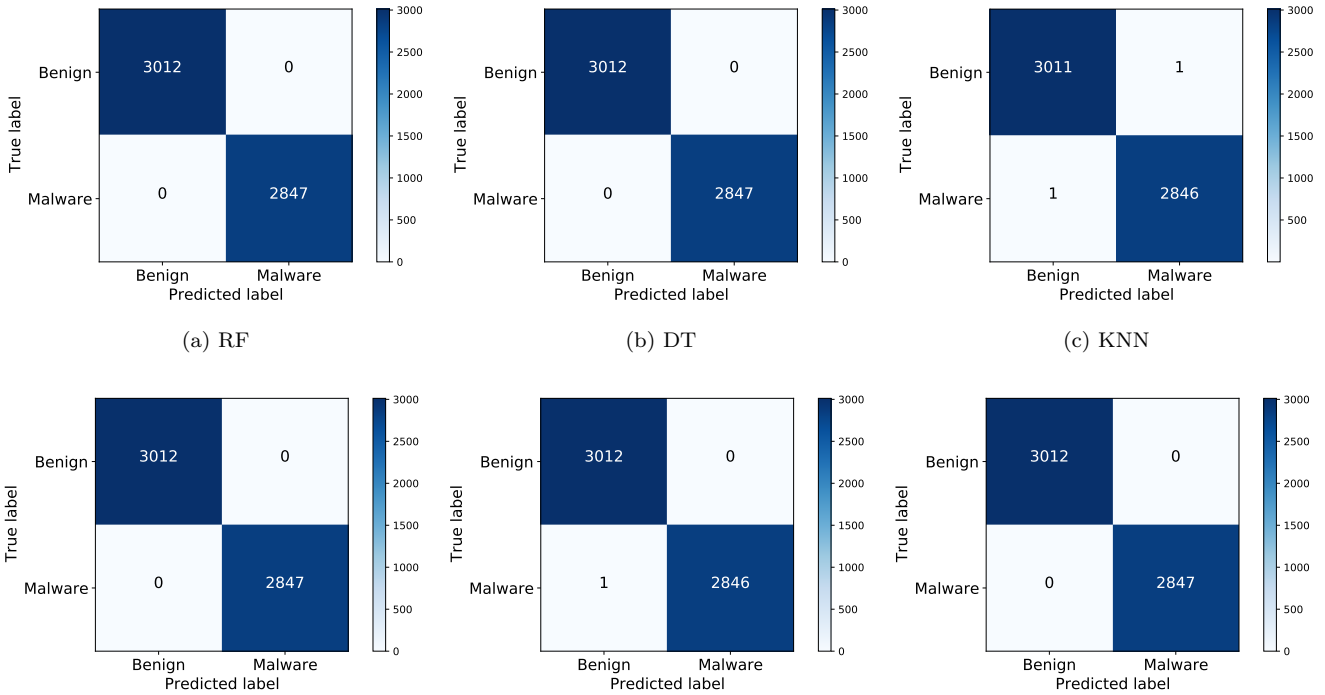


(a) Performance



(b) Error

Figure 18: Performance analysis graphs for binary classification.

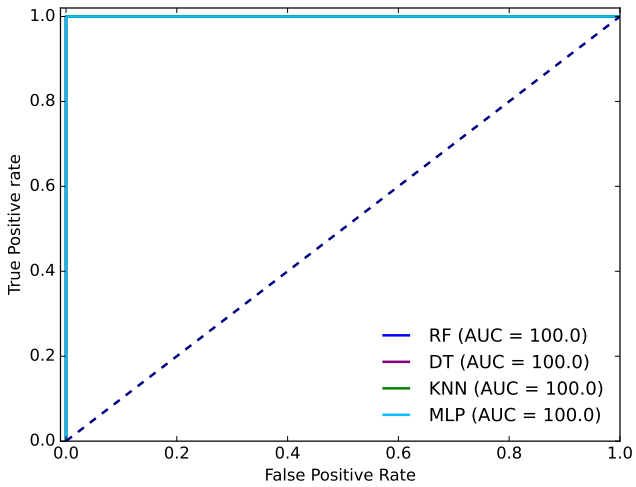


(a) RF (b) DT (c) KNN

(d) MLP (e) CNN (f) ANN

Figure 19: Confusion matrix for multilabel classification of CIC-MalMem-2022.

matrix results show that RF, DT, KNN, and MLP outperform the other algorithms using 20 features. The TP and TN rates are very high, and FN and FN rate is zero for these algorithms, which provides a better confusion matrix for accurately detecting malware memory threats.



(a) ROC Curve

Figure 20: ROC Curve for CIC-MalMem-2022.

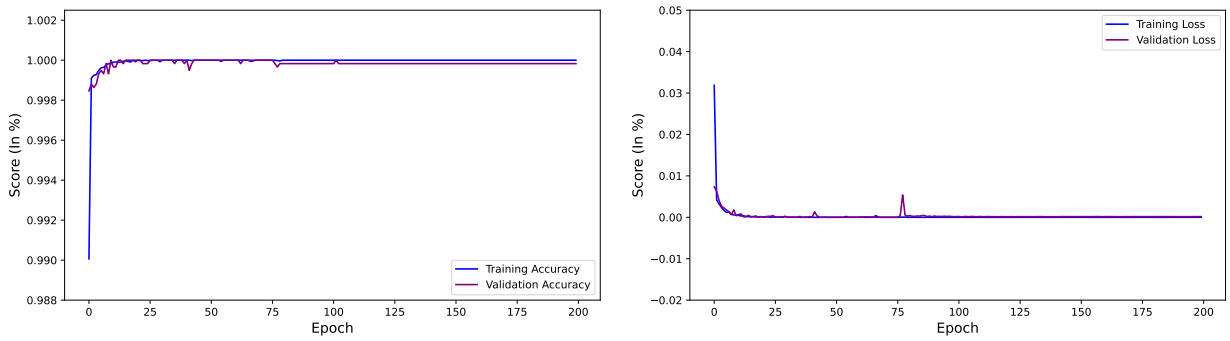
Fig 20 shows the Binary ROC Curve for CIC-MalMem-2022, where the AUC score for RF, DT, KNN, and MLP is 100%. Here, all the ML performs at the highest accuracy rate. Random Forest produces superior findings,

operates well on huge datasets, and can create estimates for missing data. It is a collection of independent decision trees collaborating to form an ensemble. As a result, knowledge can be used to train several learner algorithms to reach maximum accuracy. DT is simple to use, comprehend and explain; it takes minimal time and effort to plan and produce; it may be used in conditional probability-based reasoning and can offer strategic responses to uncertain situations. KNN enables nonlinear solutions and performs instance-based learning. A well-tuned K can model large decision spaces with arbitrarily convoluted decision boundaries that are difficult to model by other "eager" learners who work in batches, modelling one group of early phases at a time. MLP consists of a series of layers made up of neurons and their connections. It has one or more hidden layers between the input and output layers where the neurons are arranged in layers, and connections are often guided from lower to upper layers. Hence, they give better performance in our proposed model.

After analyzing all the ML and DL algorithms, we select the RF and ANN to detect network intrusion for binary classifications.

5.2.1. Dependability Analysis of our proposed approach

This section examines the effectiveness study of our developed model's dependability. The aspects of availability, efficiency, and scalability are all considered in the reliability evaluation study. We choose features based on accuracy efficiency rate and then use our developed framework to reliably differentiate between benign and attack or malware scenarios without experiencing any loss, ensuring that our presented methodology remains available. Furthermore, comprehensive analysis and productivity assessments such as accuracy, precision, recall, f1-score, AUC score, ROC Curve, MAE, MSE, and RMSE indicate that the proposed framework is more efficient and has revealed improved results than many other existing techniques with a lower error rate and computational loss.

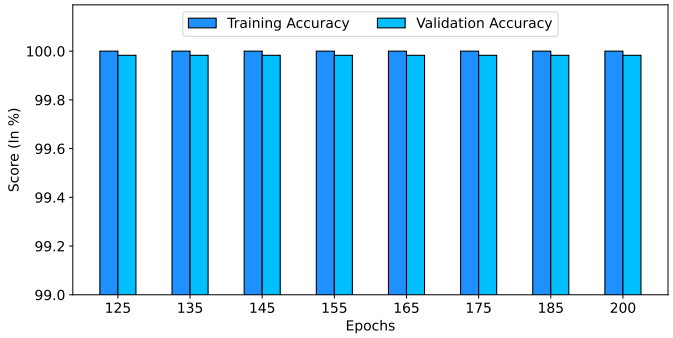


(a) Accuracy (b) Loss

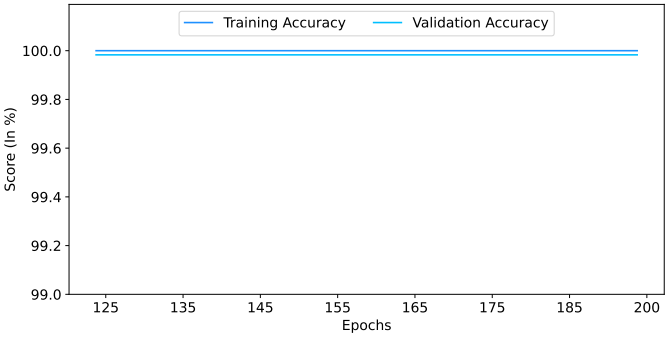
Figure 21: The effectiveness and computational loss.

Figure 21 depicts the developed model's effectiveness and computational loss. Ultimately, we found that incorporating numerous incompatible reliable datasets in the training sample with optimum reliability, which were gathered from a broader variety of IoT sensors, improved the scalability qualities of our developed framework. Consequently, when we extended the epoch quantity from 125 to 200, our proposed approach's accuracy performance rate remained nearly unchanged, confirming its flexibility or scalability.

Figure 22 illustrates the proposed model's scalability.



(a) Bar chart



(b) Line chart

Figure 22: The scalability of our proposed model.

5.2.2. Discussion

The comparison analysis with existing research work and our proposed model is shown in Table 13and Table 14 for KDDCUP'99 and CIC-MalMem-2022 datasets, respectively. The comparison results of KDDCUP'99 show that our proposed model provides outperforms other existing works considering both binary and multilabel classification. Similarly, for the CIC-MalMem-2022 dataset, the comparison results with existing work prove that our proposed model provides a better performance accuracy rate to detect malware memory attacks.

Hence, the performance results of the proposed model are compared to the recent research work ensuring that our proposed model gives a higher detection rate as well as provides the robustness and effectiveness to detect intrusion. Moreover, the aforementioned results showed that proper data preparation could assist to improve the accuracy of ML and DL algorithms, while the XGBoost technique can quickly discover the most promising features with greater accuracy outcomes.

Our proposed model is compatible with both contemporary network topologies with NIDS system and also that enable intelligently centralised network management, such as software-defined networks (SDN). In SDN, the application layer consists of network applications associated with data security, where we can employ our proposed IDS model [84]. It, therefore, can interact with network resources via the Northbound Interface (NBI) to defend the infrastructure. Control layer SDN operates as a centralised SDN controller hosted on a server to manage network traffic and policy [85]. This controller is based on the model used in the application layer. By customising the network switch's security rule, malicious traffic can be regulated to meet the needs of different business networks

SL.No.	Author	Feature Selection Method	Classification Algorithm	Selected Features	Performance (Accuracy In %)
1	[13]	-	SMOTE+RF	All	92.57
2	[16]	-	XGBoost	All	99.95
3	[18]	-	DNN	All	91.50
4	[22]	-	CNN + LSTM	All	98.48
5	[23]	-	DT	All	94.00
6	[33]	-	EMRFT	All	96.56
7	[19]	PSO	ANN	20	98.00
8	[20]	CR	DNN	30	99.40
9	[21]	BAT	SVM	25	94.12
10	[28]	FGCC+CFA	DT	10	95.03
11	[29]	IGR+CR+ ReF+SCS	PART	12	99.32
12	[30]	EFS	RF	15	93.90
13	[31]	PSO	NB	38	99.12
14	Proposed Method (Binary)	SMOTE + XGBoost	RF	20	99.99
15	Proposed Method (Multilabel)	SMOTE + XGBoost	RF	20	99.99

Table 13: Comparison analysis for KDDCUP'99 dataset.

SL.No.	Author	Dataset	Feature Selection Method	Classification Algorithm	Selected Features	Performance (Accuracy In %)
1	[35]	CIC-MalMem-2022	-	Stacked Ensemble	-	99.00
2	[36]	CIC-MalMem-2022	-	LR	-	99.97
3	[37]	CIC-MalMem-2022	-	RF	-	100
4	Proposed Method	CIC-MalMem-2022	XGBoost	RF, ANN	20	100 (RF), 100 (ANN)

Table 14: Comparison analysis for the CIC-MalMem-2022 dataset.

with little effort. This permits flexible and fruitful regulation of network congestion. Our model will make SDN highly adaptable, manageable, flexible, and cost-effective. Therefore, it can be used in both high-bandwidth and dynamic systems for network security.

One dataset in this paper is concerned with a network attack (KDDCUP'99), while the other is concerned with a malware memory attack (CIC-MalMem-2022), so their characteristics couldn't be more dissimilar. With 44 features, the KDDCUP'99 dataset contains 494021 data points. The dataset is multiclass labeled and has some imbalances. In contrast, the CIC-MalMem-2022 dataset includes 58,596 data points across 57 features. There is a binary class label in this balanced dataset.

If we compare the proposed XGBoost-based feature selection process to others like PSO, BAT, IGR, CRC, ReF, EFS, and FGLCC+CFA, etc., we can find that it provides more value. When compared to XGBoost, the current feature selection process doesn't make use of its second derivative to reduce model error and L1 & L2 regularisation to make it more generic [54, 86]. Furthermore, it's performance is enhanced by the stochastic gradient boosting algorithm [87]. Therefore, XGBoost aids in achieving superior feature selection capabilities, which in turn boost the model's performance more than competing methods.

6. Conclusion

The purpose of this study is to develop a novel, dependable, and effective network intrusion detection system. In order to achieve that, we developed a hybrid machine learning approach that included data balancing with SMOTE and the extraction of dominant features with XGBoost. ML and DL algorithms like RF, DT, KNN, MLP, CNN, and ANN were used to test and evaluate the proposed method in order to find the best model for detecting network intrusion. Several performance metrics were used to determine how well the binary and multiclass attack algorithms worked. The performance results show that among all ML and DL algorithms, RF has the highest accuracy rate of 99.99% with the chosen features for the KDDCUP'99 dataset and 100% for the CIC-MalMem-2022 dataset. Furthermore, performance comparison with previous research demonstrates the model's dependability and robustness over others.

Overall, the key findings of this paper are summarized as follows:

- Our proposed hybrid model performs significantly better than traditional ML and DL-based models because it integrates effective preprocessing, feature scaling, and feature selection.
- In the traditional approach, often one class predicts more accurately than others due to data imbalance, resulting in an imbalanced classification model. SMOTE data balance, in our approach, helps produce a better prediction model by removing the negative effect of class imbalance.
- SMOTE helps balance the dataset, which leads to a more accurate prediction model by reducing type-1 (false positive) and type-2 (false negative) errors. In general, the negative effect of a model's performance due to the data imbalance is clear when examining the confusion matrix. When true positives and negatives are larger than false positives and false negatives, this demonstrates that the model can only predict a particular class label. If new data is tested on the class imbalance model, performance will be significantly reduced due to this issue. Using SMOTE in our approach reduces type-1 and type-2 errors.
- The use of XGBoost extracts dominant features to improve the accuracy. XGBoost uses extra precise approximate to find the best model tree. It also uses the second derivative to minimize model error and L1 & L2 regularization to generalize it. The stochastic gradient boosting algorithm within it helps to improve accuracy. XGBoost also has an inbuilt process to get feature importance for feature selection by relating multiple variables to feature importance. Overall, it contributes to achieving excellent feature selection capabilities to improve the model's performance.
- Such a hybrid pipeline can reliably offer a superior detection rate along with the model's availability and scalability.

Overall, our proposed approach is dependable and can be implemented in real-time by installing this model on internet-connected IDS devices. In future, we will investigate our model's performance with any emerging threats upon the availability of the latest datasets. To further enhance the performance of the intrusion detection system, we will also compare our model's results with those of the ensemble feature selection method, which uses the union and intersection of features. This will assist in identifying relationships for selecting the most important features to use in the deep neural network.

References

[1] 2020 Trustwave Global Security Report;. [Online; accessed 2022-09-13]. https://www.trustwave.com/en-us/resources/library/documents/2020-trustwave-global-security-report/.

[2] Sarker IH, Kayes A, Badsha S, Alqahtani H, Watters P, Ng A. Cybersecurity data science: an overview from machine learning perspective. Journal of Big data. 2020;7(1):1-29.

[3] Dash S, Shakyawar SK, Sharma M, Kaushik S. Big data in healthcare: management, analysis and future prospects. Journal of Big Data. 2019;6(1):1-25.

[4] Hasan KF, Overall A, Ansari K, Ramachandran G, Jurdak R. Security, privacy and trust: cognitive internet of vehicles. arXiv preprint arXiv:210412878. 2021.

[5] Wei W, Guo C. A text semantic topic discovery method based on the conditional co-occurrence degree. Neurocomputing. 2019;368:11-24.

[6] Humayn Kabir M, Fida Hasan K, Kamrul Hasan M, Ansari K. Explainable Artificial Intelligence for Smart City Application: A Secure and Trusted Platform. arXiv e-prints. 2021:arXiv-2111.

[7] Mandal K, Rajkumar M, Ezhumalai P, Jayakumar D, Yuvarani R. Improved security using machine learning for IoT intrusion detection system. Materials Today: Proceedings. 2020.

[8] Hasan KF, Kaur T, Hasan MM, Feng Y. Cognitive internet of vehicles: motivation, layered architecture and security issues. In: 2019 International Conference on Sustainable Technologies for Industry 4.0 (STI). IEEE; 2019. p. 1-6.

[9] Tomar D, Tomar P. Dimensionality reduction techniques for IoT based data. Recent Advances in Computer Science and Communications (Formerly: Recent Patents on Computer Science). 2021;14(3):724-35.

[10] Ayesha S, Hanif MK, Talib R. Overview and comparative study of dimensionality reduction techniques for high dimensional data. Information Fusion. 2020;59:44-58.

[11] Garg S, Kaur K, Kumar N, Batra S, Obaidat MS. HyClass: Hybrid classification model for anomaly detection in cloud environment. In: 2018 IEEE International Conference on Communications (ICC). IEEE; 2018. p. 1-7.

[12] Li X, Yi P, Wei W, Jiang Y, Tian L. LNNLS-KH: a feature selection method for network intrusion detection. Security and Communication Networks. 2021;2021.

[13] Tan X, Su S, Huang Z, Guo X, Zuo Z, Sun X, et al. Wireless sensor networks intrusion detection based on SMOTE and the random forest algorithm. Sensors. 2019;19(1):203.

[14] Ahmed HA, Hameed A, Bawany NZ. Network intrusion detection using oversampling technique and machine learning algorithms. PeerJ Computer Science. 2022;8:e820.

[15] Gonzalez-Cuautle D, Hernandez-Suarez A, Sanchez-Perez G, Toscano-Medina LK, Portillo-Portillo J, Olivares-Mercado J, et al. Synthetic minority oversampling technique for optimizing classification tasks in botnet and intrusion-detection-system datasets. Applied Sciences. 2020;10(3):794.

[16] Bhati BS, Chugh G, Al-Turjman F, Bhati NS. An improved ensemble based intrusion detection technique using XGBoost. Transactions on Emerging Telecommunications Technologies. 2021;32(6):e4076.

[17] Nimbalkar P, Kshirsagar D. Feature selection for intrusion detection system in Internet-of-Things (IoT). ICT Express. 2021;7(2):177-81.

[18] Choudhary S, Kesswani N. Analysis of KDD-Cup'99, NSL-KDD and UNSW-NB15 Datasets using Deep Learning in IoT. Procedia Computer Science. 2020;167:1561-73.

[19] Norwahidayah S, Farahah N, Amirah A, Liyana N, Suhana N, et al. Performances of Artificial Neural Network (ANN) and Particle Swarm Optimization (PSO) Using KDD Cup '99 Dataset in Intrusion Detection System (IDS). In: Journal of Physics: Conference Series. vol. 1874. IOP Publishing; 2021. p. 012061.

[20] Li LH, Ahmad R, Tsai WC, Sharma AK. A Feature Selection Based DNN for Intrusion Detection System. In: 2021 15th International Conference on Ubiquitous Information Management and Communication (IMCOM). IEEE; 2021. p. 1-8.

[21] Narayanasami S, Sengan S, Khurram S, Arslan F, Murugaiyan SK, Rajan R, et al. Biological Feature Selection and Classification Techniques for Intrusion Detection on BAT. Wireless Personal Communications. 2021:1-23.

[22] Hu Y, Liu R, Ma Z. Identification of Cybersecurity Elements Based on Convolutional Attention LSTM Networks. In: Journal of Physics: Conference Series. vol. 1757. IOP Publishing; 2021. p. 012146.

[23] Alqahtani H, Sarker IH, Kalim A, Hossain SMM, Ikhlaq S, Hossain S. Cyber intrusion detection using machine learning classification techniques. In: International Conference on Computing Science, Communication and Security. Springer; 2020. p. 121-31.

[24] Kumar V, Das AK, Sinha D. Statistical analysis of the UNSW-NB15 dataset for intrusion detection. In: Computational Intelligence in Pattern Recognition. Springer; 2020. p. 279-94.

[25] Koroniotis N, Moustafa N, Sitnikova E, Slay J. Towards developing network forensic mechanism for botnet activities in the iot based on machine learning techniques. In: International Conference on Mobile Networks and Management. Springer; 2017. p. 30-44.

[26] Kasongo SM, Sun Y. Performance analysis of intrusion detection systems using a feature selection method on the UNSW-NB15 dataset. Journal of Big Data. 2020;7(1):1-20.

[27] Salo F, Nassif AB, Essex A. Dimensionality reduction with IG-PCA and ensemble classifier for network intrusion detection. Computer Networks. 2019;148:164-75.

[28] Mohammadi S, Mirvaziri H, Ghazizadeh-Ahsaee M, Karimipour H. Cyber intrusion detection by combined feature selection algorithm. Journal of information security and applications. 2019;44:80-8.

[29] Kshirsagar D, Kumar S. An efficient feature reduction method for the detection of DoS attack. ICT Express. 2021.

[30] Mugabo E, Zhang QY, Ngaboyindekwe A, Kwizera VdPN, Lumorvie VE. Intrusion Detection Method Based on MapReduce for Evolutionary Feature Selection in Mobile Cloud Computing. International Journal of Network Security. 2021;23(1):106-15.

[31] Talita A, Nataza O, Rustam Z. Naive Bayes Classifier and Particle Swarm Optimization Feature Selection Method for Classifying Intrusion Detection System Dataset. In: Journal of Physics: Conference Series. vol. 1752. IOP Publishing; 2021. p. 012021.

[32] Zhao F, Zhao J, Niu X, Luo S, Xin Y. A filter feature selection algorithm based on mutual information for intrusion detection. Applied Sciences. 2018;8(9):1535.

[33] Mahhizharuvi P, et al. An Effective Intrusion Detection System using Enhanced Multi relational Fuzzy Tree. Turkish Journal of Computer and Mathematics Education (TURCOMAT). 2021;12(13):3152-9.

[34] Indrasiri PL, Lee E, Rupapara V, Rustam F, Ashraf I. Malicious traffic detection in iot and local networks using stacked ensemble classifier. Computers, Materials and Continua. 2022;71(1):489-515.

[35] Carrier T, Victor P, Tekeoglu A, Lashkari A. Detecting Obfuscated Malware using Memory Feature Engineering. In: Proceedings of the 8th International Conference on Information Systems Security and Privacy - ICISSP,. INSTICC. SciTePress; 2022. p. 177-88.

[36] Dener M, Ok G, Orman A. Malware Detection Using Memory Analysis Data in Big Data Environment. Applied Sciences. 2022;12(17):8604.

[37] Louk MHL, Tama BA. Tree-Based Classifier Ensembles for PE Malware Analysis: A Performance Revisit. Algorithms. 2022;15(9):332.

[38] Man J, Sun G. A residual learning-based network intrusion detection system. Security and Communication Networks. 2021;2021.

[39] Shetty NP, Shetty J, Narula R, Tandona K. Comparison study of machine learning classifiers to detect anomalies. International Journal of Electrical and Computer Engineering. 2020;10(5):5445.

[40] Chawla NV, Bowyer KW, Hall LO, Kegelmeyer WP. SMOTE: synthetic minority over-sampling technique. Journal of artificial intelligence research. 2002;16:321-57.

[41] Rustam F, Mehmood A, Ullah S, Ahmad M, Khan DM, Choi GS, et al. Predicting pulsar stars using a random tree boosting voting classifier (RTB-VC). Astronomy and Computing. 2020;32:100404.

[42] He H, Garcia EA. Learning from imbalanced data. IEEE Transactions on knowledge and data engineering. 2009;21(9):1263-84.

[43] Barandela R, Valdovinos RM, Sánchez JS, Ferri FJ. The imbalanced training sample problem: Under or over sampling? In: Joint IAPR international workshops on statistical techniques in pattern recognition (SPR) and structural and syntactic pattern recognition (SSPR). Springer; 2004. p. 806-14.

[44] Motoda H, Liu H. Feature selection, extraction and construction. Communication of IICM (Institute of Information and Computing Machinery, Taiwan). 2002;5(67-72):2.

[45] Wei H. Feature Selection Methods with Code Examples — by Haitian Wei — Analytics Vidhya — Medium; 2019. Accessed: 2022-02-19. https://www.analyticsvidhya.com/blog/2020/10/feature-selection-techniques-in-machine-learning/.

[46] Tang Y, Zhao Z, Zhang S, Li Z, Mo Y, Guo Y. Motor Imagery EEG Decoding Based on New Spatial-Frequency Feature and Hybrid Feature Selection Method. Mathematical Problems in Engineering. 2022;2022.

[47] Kharwar AR, Thakor DV. An Ensemble Approach for Feature Selection and Classification in Intrusion Detection Using Extra-Tree Algorithm. International Journal of Information Security and Privacy (IJISP). 2022;16(1):1-21.

[48] Mojtahedi A, Sorouri F, Souha AN, Molazadeh A, Mehr SS. Feature Selection-based Intrusion Detection System Using Genetic Whale Optimization Algorithm and Sample-based Classification. arXiv preprint arXiv:220100584. 2022.

[49] Vaidya A, Kshirsagar D. Analysis of Feature Selection Techniques to Detect DoS Attacks Using Rule-Based Classifiers. In: Applied Information Processing Systems. Springer; 2022. p. 311-9.

[50] Devan P, Khare N. An efficient XGBoost–DNN-based classification model for network intrusion detection system. Neural Computing and Applications. 2020:1-16.

[51] John Lu Z. The elements of statistical learning: data mining, inference, and prediction. Wiley Online Library; 2010.

[52] Farrugia S, Ellul J, Azzopardi G. Detection of illicit accounts over the Ethereum blockchain. Expert Systems with Applications. 2020;150:113318.

[53] Dhaliwal SS, Nahid AA, Abbas R. Effective intrusion detection system using XGBoost. Information. 2018;9(7):149.

[54] Chen T, Guestrin C. Xgboost: A scalable tree boosting system. In: Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining; 2016. p. 785-94.

[55] Alkhatib K, Abualigah S. Predictive Model for Cutting Customers Migration from banks: Based on machine learning classification algorithms. In: 2020 11th International Conference on Information and Communication Systems (ICICS). IEEE; 2020. p. 303-7.

[56] Breiman L. Random forests. Machine learning. 2001;45(1):5-32.

[57] Ahmad M, Riaz Q, Zeeshan M, Tahir H, Haider SA, Khan MS. Intrusion detection in internet of things using supervised machine learning based on application and transport layer features using UNSW-NB15 data-set. EURASIP Journal on Wireless Communications and Networking. 2021;2021(1):1-23.

[58] Pan X, Zhu L, Fan YX, Yan J. Predicting protein–RNA interaction amino acids using random forest based on submodularity subset selection. Computational biology and chemistry. 2014;53:324-30.

[59] Gavankar SS, Sawarkar SD. Eager decision tree. In: 2017 2nd International Conference for Convergence in Technology (I2CT). IEEE; 2017. p. 837-40.

[60] Dey A. Machine learning algorithms: a review. International Journal of Computer Science and Information Technologies. 2016;7(3):1174-9.

[61] Mrva J, Neupauer Š, Hudec L, Ševcech J, Kapec P. Decision Support in Medical Data Using 3D Decision Tree Visualisation. In: 2019 E-Health and Bioengineering Conference (EHB). IEEE; 2019. p. 1-4.

[62] Ahmed N, Ahammed R, Islam MM, Uddin MA, Akhter A, Talukder MAA, et al. Machine learning based diabetes prediction and development of smart web application. International Journal of Cognitive Computing in Engineering. 2021;2:229-41.

[63] Jahan S, Islam M, Islam L, Rashme TY, Prova AA, Paul BK, et al. Automated invasive cervical cancer disease detection at early stage through suitable machine learning model. SN Applied Sciences. 2021;3(10):1-17.

[64] Castro W, Oblitas J, Santa-Cruz R, Avila-George H. Multilayer perceptron architecture optimization using parallel computing techniques. PloS one. 2017;12(12):e0189369.

[65] Ramchoun H, Idrissi MAJ, Ghanou Y, Ettaouil M. Multilayer Perceptron: Architecture Optimization and Training. IJIMAI. 2016;4(1):26-30.

[66] Talukder MA, Islam MM, Uddin MA, Akhter A, Hasan KF, Moni MA. Machine learning-based lung and colon cancer detection using deep feature extraction and ensemble learning. Expert Systems with Applications. 2022:117695.

[67] Valueva MV, Nagornov N, Lyakhov PA, Valuev GV, Chervyakov NI. Application of the residue number system to reduce hardware costs of the convolutional neural network implementation. Mathematics and Computers in Simulation. 2020;177:232-43.

[68] Zhang W, Tanida J, Itoh K, Ichioka Y. Shift-invariant pattern recognition neural network and its optical architecture. In: Proceedings of annual conference of the Japan Society of Applied Physics. Montreal, CA; 1988. p. 2147-51.

[69] Zhang W, Itoh K, Tanida J, Ichioka Y. Parallel distributed processing model with local space-invariant interconnections and its optical architecture. Applied optics. 1990;29(32):4790-7.

[70] Fukushima K, Miyake S. Neocognitron: A self-organizing neural network model for a mechanism of visual pattern recognition. In: Competition and cooperation in neural nets. Springer; 1982. p. 267-85.

[71] Yamashita R, Nishio M, Do RKG, Togashi K. Convolutional neural networks: an overview and application in radiology. Insights into imaging. 2018;9(4):611-29.

[72] Matsugu M, Mori K, Mitari Y, Kaneda Y. Subject independent facial expression recognition with robust face detection using a convolutional neural network. Neural Networks. 2003;16(5-6):555-9.

[73] Lin M, Chen Q, Yan S. Network in network. arXiv preprint arXiv:13124400. 2013.

[74] Arvinth S, Balakrishnan A, Harikrishnan M, Jeydheepan J. Weed Detection Using Convolution Neural Network. International Research Journal of Modernization in Engineering Technology and Science. 2021;3(3).

[75] Ahamed KU, Islam M, Uddin A, Akhter A, Paul BK, Yousuf MA, et al. A deep learning approach using effective preprocessing techniques to detect COVID-19 from chest CT-scan and X-ray images. Computers in biology and medicine. 2021;139:105014.

[76] Bland C, Tonello L, Biganzoli E, Snowdon D, Antuono P, Lanza M. Advances in Artificial Neural Networks. Advances in Artificial Neural Networks. 2020:119.

[77] Gorgun E. Characterization of Superalloys by Artificial Neural Network Method. In: Online International Symposium on Applied Mathematics and Engineering (ISAME22) January 21-23, 2022 Istanbul-Turkey; 2022. p. 67.

[78] Feldmann J, Youngblood N, Wright CD, Bhaskaran H, Pernice WH. All-optical spiking neurosynaptic networks with self-learning capabilities. Nature. 2019;569(7755):208-14.

[79] Hossain MU, Rahman MA, Islam MM, Akhter A, Uddin MA, Paul BK. Automatic driver distraction detection using deep convolutional neural networks. Intelligent Systems with Applications. 2022;14:200075.

[80] DARPA. KDD-CUP 1999 Dataset, US Department of Defense Advanced Research Projects Agency (DARPA); 1999. Accessed: 2021-02-21. `http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html`.

[81] Zhong M, Zhou Y, Chen G. Sequential model based intrusion detection system for IoT servers using deep learning methods. Sensors. 2021;21(4):1113.

[82] Siddique K, Akhtar Z, Khan FA, Kim Y. KDD Cup 99 data sets: a perspective on the role of data sets in network intrusion detection research. Computer. 2019;52(2):41-51.

[83] Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, et al. Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research. 2011;12:2825-30.

[84] Li W, Wang Y, Jin Z, Yu K, Li J, Xiang Y. Challenge-based collaborative intrusion detection in software-defined networking: An evaluation. Digital Communications and Networks. 2021;7(2):257-63.

[85] Yan Q, Yu FR, Gong Q, Li J. Software-defined networking (SDN) and distributed denial of service (DDoS) attacks in cloud computing environments: A survey, some research issues, and challenges. IEEE communications surveys & tutorials. 2015;18(1):602-22.

[86] Wang C, Deng C, Wang S. Imbalance-XGBoost: leveraging weighted and focal losses for binary label-imbalanced classification with XGBoost. Pattern Recognition Letters. 2020;136:190-7.

[87] Kiangala SK, Wang Z. An effective adaptive customization framework for small manufacturing plants using extreme gradient boosting-XGBoost and random forest ensemble learning algorithms in an Industry 4.0 environment. Machine Learning with Applications. 2021;4:100024.