Journal of Computer Science

Research Paper

Online ML Streaming-Based Leading Ensemble Classifier for Intrusion Detection in IoT

Neeraj Sharma[1], Neelu Nihalani[2]

[1]*Department of Computer Science & Engineering, University Institute of Technology, Rajiv Gandhi Proudyogiki Vishwavidyalaya, Bhopal, Madhya Pradesh, India;*
[2]*Department of Computer Applications, University Institute of Technology, Rajiv Gandhi Proudyogiki Vishwavidyalaya, Bhopal, Madhya Pradesh, India;*

Article history
Received: 10 February 2025
Revised: 14 April 2025
Accepted: 20 May 2025

Corresponding Author:
Neeraj Sharma, Department of Computer Science & Engineering, University Institute of Technology, Rajiv Gandhi Proudyogiki Vishwavidyalaya, Bhopal, Madhya Pradesh, India
Email: neerajsharmans12@gmail.com

Abstract: In the present decade, preventing IoT networks from malicious attacks and to ensure the privacy and integrity of data by strengthening their cybersecurity perspective is a critical concern. Traditional methods alone are not sufficient to detect multiple attacks. IDS-based system plays a crucial role to maintain security due to continuous expanding nature of IoT networks. Traditional methods alone are often insufficient to detect and counter attacks that try to impact IoT devices. This creates a demand for advanced technological approaches to boost the security of IoT networks. One approach gaining attraction in addressing this challenge is by deploying ensemble machine learning models for identifying the attack categories and preventing them. To build an ensemble framework for the identification of malicious behavior in IoT-based network, this study combines the results and metrics obtained from 4 base classifiers LightGBM, Random forest classifier (RFC), XGBoost, and CatBoost. Merged them into an online ML streaming-based Leading Ensemble Decision Classifier Module (LEDCM). This module predicts the accurate class for every record in both multiclass and binary classification scenarios using IoT-based datasets (i.e. NSL-KDD, NF-BOT-IoT, UNSW-NB-15 datasets), LEDCM merge the prediction from all the major 4 classifiers in such a way, the overall Accuracy % and f1-score will get increased by fetching the best prediction from all base classifiers in a group individually Class wise. In the final results, we observed that our proposed model achieved the highest accuracy of 99.94% for NSL-KDD test data, 95.99% for UNSW-NB-15, and 83.13% for NF-BOT-IoT for Multiclass classification, and Accuracy of 100% for all three Datasets in the case of binary label classification in comparison to other base classifiers.

Keywords: Cybersecurity, IDS, Feature selection, Ensemble, Anomaly detection, ML, IoT, Attacks

Introduction

The exponential growth of Internet of Things (IoT) scenarios has led to millions of interconnected data transmission devices and the generation of massive quantities of data. IoT offers a lot of advantages and benefits, but it has increased the risk of fraud as well because it allows it to penetrate and impact the data. Information will be at risk because we may have many open doors through which data can be accessed (Alanazi et al., 2015). Reliable Intrusion Detection Systems (IDS) play a crucial role in observing, detecting, stopping, and preventing breaches in IoT networks (Granjal et al., 2015). Traditional IDS methods are not capable of working strongly enough in the case of IoT. Traditional IDS methods, such as signature-based detection, are often insufficient for addressing the sophisticated and changing nature of IoT attacks. As a result, the need for more advanced and efficient IDS models has emerged. In this work, we propose the creation and application of a framework for an ensemble approach for accurate and reliable detection

SCIENCE Publications

of breach activity; it can be achieved by developing efficient intrusion detection systems for IoT-based networks. The ensemble model combines works of multiple IDS techniques to reduce the impact of false positives or negatives, such that it will be reduced during the prediction mechanism that helps in minimizing the impact of risks that may be generated or faced in the case of individual model scenarios. By leveraging the strengths of different detection methods, the ensemble model aims to maximize the performance of intrusion detection systems.

By combining the predictions of multiple models, the ensemble model can effectively identify breaches and distinguish between normal and malicious activities within IoT networks. This approach leverages the collective intelligence of the models that will help to minimize the risk of false positives as well as the risk of false negatives that individual models might produce. To deploy an ensemble framework for attack detection in the case of IoT-based networks, a diverse set of features extracted or obtained from network traffic data and device logs can be used. These features may include network packet statistics, traffic flow characteristics, behavior patterns of devices, and communication patterns between devices.

The ensemble model is trained on labeled datasets where both normal and attack instances are included. The training stage consists of optimizing the model's parameters and determining the combination of individual models that yields the best performance. Once the ensemble framework is trained on datasets, it can be made operational in real-time scenarios of IoT networks to detect and respond to attacks. Incoming network traffic and device data are continuously monitored, and the ensemble model evaluates the data to identify any suspicious patterns or anomalies. If an attack is detected, then in response, appropriate actions can be taken, such as blocking malicious traffic or isolating affected devices. The benefits we have of using an ensemble framework of machine learning classifiers for breach detection in cybersecurity is their ability to gain or increase the overall performance, accuracy, and robustness of the framework or model. With a combination of different models, the ensemble framework can better generalize patterns and adapt to evolving attack techniques. In the case of Unbalanced datasets, where we see huge variations in several records between multiple class labels (Attack types), ensemble frameworks will outperform the capabilities of Individual classifiers. In this study, we propose a hybrid framework that merges the capabilities of multiple machine learning algorithms to create an accurate and precise intrusion detection system for networks. The proposed model leverages the strengths of individual algorithms to achieve an enhanced detection performance. This study also provides a good understanding of multiple classification algorithms used to prevent breaches over the IoT datasets, the major aim is to develop a combined ensemble IDS framework (Zarpelao et al.,2017) (i.e. that is a collection of weak classifiers into to strong single classifier model), the complete framework provides a good description about the steps we followed for individual classifier implementation as well as their combined model implementation validation procedures, and deployment strategies.

Motivation

Attack detection in cybersecurity for IoT networks is a complex challenge that requires sophisticated technologies. Ensemble machine learning frameworks ensure a promising mechanism to increase the security level of the network traffic of IoT, by effectively detecting and responding to attacks. With further advancements and research in this field, ensemble models have great potential to ensure the privacy and integrity of traffic in IoT networks in our increasingly interconnected world. The benefit we have of using an ensemble framework of machine learning classifiers or models for breach or malicious intent detection in cybersecurity is their ability to gain or increase the overall performance, accuracy, and robustness of the framework or model. With a combination of different models, the ensemble framework can better generalize patterns and adapt to evolving attack techniques. In the case of Unbalanced datasets, where we see huge variations in the number of records between multiple class labels (Attack types), ensemble frameworks will outperform the capabilities of Individual classifiers.

a) Designing and implementing a diverse set of intrusion detection techniques: As we know ensemble model works in a way by combining multiple machine learning algorithms and every algorithm has its rich strength in multiple scenarios, so in overall consideration, these diverse sets of ensemble machine learning algorithms will provide great results, because wherever a Classifier-A underperform, we will consider other classifier- B that have good performance and we have a choice to make a selection in ensemble design.
b) Developing an ensemble framework: Every ensemble model has a diverse range of base classifier algorithms and they may have a different way of working let’s say some have different ways of adjusting weight factors or weighting mechanisms, voting-based predictions, etc, and different ways to examine network traffics or Individual flow, so in our ensemble-based model, none of the good probability will get ignored instead that will be considered.

c) Evaluating the performance of the ensemble model: The overall models will be trained on real-life datasets, and then after the predictions are made, we will see the overall gain in terms of metrics attributes, such as recall% %, accuracy, F-score, etc.

Novelty and Contribution

In this aspect, we designed an Online ML Streaming framework of an Incremental nature (i.e., data will be evaluated consecutively instead of batches) that is different from batch-processing/offline-based learning frameworks. Online ML frameworks are capable of handling issues related to scale and stationarity in data. We proposed an LEDCM module that is different from other forms of ensemble techniques, i.e., bagging, boosting, stacking, voting, etc.

a) It supports the heterogeneous collection of weak learners in comparison to bagging and boosting, which support only homogeneous collections.

b) It does not require any meta-model feed-up step as in the case of stacking. Instead of multiple passes, it performs only a single pass over the data collectively.

c) LEDCM supports an online Machine learning Approach, in which data becomes available in sequential order and is used to update the best predictor for future data at each step using the concept of Leader Model identification class-wise priori using base learners.

d) The necessity for accommodating all data within the memory becomes a limiting factor. Achieving this can pose challenges in numerous Big data applications, often proving difficult or even impossible, and the re-training process of the model can be notably sluggish. Accessing past data for the model presents a substantial hurdle, especially in scenarios where data is consistently generated. The storage of historical data demands a substantial infrastructure. Alternatively, models can undergo training in an "online" or streaming mode, treating data as a continuous stream or sequence of items fed to the model individually.

e) River (Montiel et al., 2021) Python library for data stream analytics, was employed to tackle concept drift using online machine learning (ML) models. The distinguishing feature of River lies in its ability to update all available learning models with a single incoming instance, enabling these methods to adapt and learn from data streams effectively.

f) LEDCM is an online ML framework that uses static classifiers as variables, using the River library (Python 3.8+) generates an online data flow sequentially, and a collective Ensemble label prediction is made dynamically by all learners on every iteration.

Intrusion Detection System

In this study, we propose an ensemble framework that merges the capabilities of multiple machine learning algorithms to create an accurate and precise intrusion or breach detection system for IoT networks. The ensemble model leverages the strengths of individual algorithms to achieve an enhanced detection performance. The ensemble model has three major steps to be followed: pre-processing of given data, feature selection of required attributes, and ensemble Algorithm learning. While IoT brings numerous advantages and convenience, it also poses significant security risks. Intrusion Detection Systems (IDS) are crucial in identifying malicious threats and mitigating malicious threats in IoT-based environments. To accomplish our goal, we have discussed numerous intrusion detection Systems (IDS) in our literature review section for preventing assaults in the IoT ecosystem, with the important aspects covered in this survey being detection technology, validation approach, and deployment strategy. In Fig. 1 we have mentioned multiple classification approaches for IDS available in literature.

Traditional Intrusion detection methods are not that capable, as they use key-based transmission or some other form of encryption and decryption methods to transfer data packets; some most common algorithms are RSA, Diffie-Hellman, etc. Machine learning techniques, on the other hand, have completely outperformed traditional Intrusion detection methods by gaining in the

Selection in intrusion detection. Dimension reduction has been shown to enhance classification efficiency while improving or maintaining classification accuracy in Intrusion Detection Systems (IDSs).

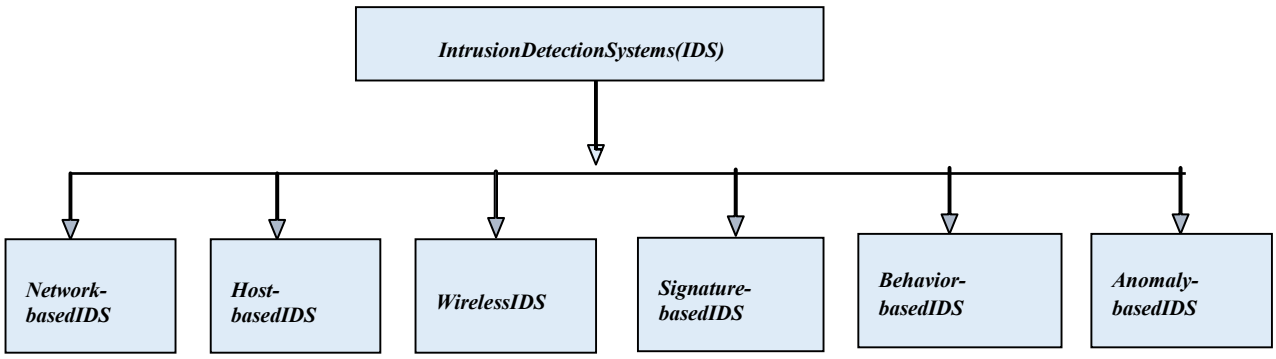


Fig. 1. Classification of IDS

Intrusion Detection System Classification

a) Network-based IDS: It works by identifying abnormal patterns in traffic flow attributes, such as the total count of Invalid attempts, src_bytes, and Invalid password attempts, etc, received from the sender to the receiver or some other type of flow.
b) Host-based IDS: Focuses on monitoring activities on individual hosts or endpoints. It examines log file system configurations and file integrity to detect unauthorized access or abnormal behavior.
c) Wireless IDS: It identifies rogue access points, detects unauthorized devices, and monitors wireless traffic for security threats. For wireless transmission scenarios, they are developed and considered. (i.e, wireless media cases).
d) Signature-based IDS: They maintain a Database of all previous attacking pattern attributes to identify intrusion. It compares network traffic or host information against the signature database to flag potential threats.
e) Behavior-based IDS: Analyzes the behavior and actions of users' network devices or host systems to identify suspicious activities. It establishes a baseline of normal behavior and alerts when deviations occur.
f) Anomaly-based IDS: Looks for deviations from normal network or host behavior that may indicate an attack. It uses statistical algorithms or machine learning techniques to establish normal patterns and detect anomalies.

Efficient IDS Techniques for IoT

We have covered some supervised machine learning. Procedures in this section. References to significant research publications are provided along with a detailed presentation of each technique.

(i) Supervised learning:

These are some of the core methods where we try to make a neural network to learn first by providing data, and we also specify the output as well (i.e., we provide the Input values and also we provide what the output should be for it).

a) Naive Bayes: The Bayes principle is used as the foundation of this technique, with strong independent hypotheses among the attributes. Naive Bayes uses conditional probability equations to answer questions like, "Exactly what was the likelihood that a specific attack type would occur, given the generated overall system behavior?" The Naive Bayes algorithm follows and works on the principle of varying probability percentage values for normal and abnormal data.

b) Genetic Algorithms: These procedures are a kind of heuristic optimization that is based on evolutionary theory. Each answer is a collection of bytes (basically, we call it genes or chromosomes in the case of the GA approach). The precision or answer accuracy gradually increases by employing biased replication and selection techniques that favor better solutions. When using genetic algorithms to address the Intrusion detection paradigms, chromosome encoding is usually classified into two types: one follows a clustering phenomenon to build a binary-encoding approach for chromosomes, while the other specifies the cluster center by an integer coding chromosome.

c) ANN: Artificial neural network: ANN is an example of machine learning methods, whose function is divided into three parts, as shown in the figure

- The first part measures the quality of output from the given input, known as the error function.
- The second part is known as the search function, whose work is to measure and decide how Much magnitude has to be changed to reduce the error function based on the results we get after applying the search function, the overall weights within the

network will be updated by the weight function

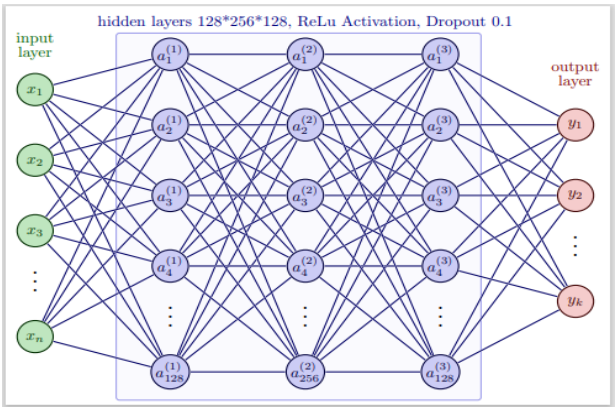


Fig. 2. Neural Network

Fig. 2 represents input nodes in the input layer referred to by a1(1) to am(n). There may be many hidden layers as per the accuracy need, and this can be achieved by adjusting weights. Also, the third layer will be an output layer in the framework.

d) Fuzzy logic: The term Fuzzy itself is the synonymy of uncertainty, when a particular data belongs to or can be represented by two or more than two classes then the term fuzzy will come into existence, it is not similar to the Boolean theory where the data can be evaluated by True or False characteristics, we can use its capability as an ML classifier for IDS scenarios, we can easily identify duplicate or ambiguous data, irrelevant, unimportant data and remove them, In fuzzy evaluation a data can belong to multiple classes simultaneously, fuzzy systems can help to identify the record in case of ambiguous behavior of data

IDS verification techniques

The process of determining if the IoT IDS model precisely represents the system to identify IoT attacks is known as IDS validation. Researchers employed a variety of approaches, such as conceptual, empirical, and hypothetical methods of authenticating their processes, to demonstrate the usefulness of IDSs. We will use the IoT datasets for further evaluation of the Ensemble model accuracy for Intrusion detection. The dataset contains normal traffic patterns as well as data for different types of attacks, such as fuzzers, DoS, worms, backdoors, etc.

Literature Review

Analyzing key studies on IoT IDSs

In the current scenarios, the deep learning mechanism plays a very crucial role in identifying patterns in complex emerging datasets with complicated representations. The hybrid model (Saini et al., 2023) contains multiple IDS techniques to reduce the impact of false positives or negatives, such that it will be reduced during the prediction mechanism that helps in minimizing the impact of risks that may be generated or faced in the case of individual model scenarios. Table 1 shows work done in the field of IoT IDS along with the limitations of each work. Many Research studies are available for Deep Learning in the literature. (Lalduhsaka et al., 2022) Proposed a Research on deep learning using data set, i.e., UNB ISCX IDS 2012. The Deep Learning Network used contains about 200 nodes for each layer, and there were about 4 hidden layers implemented for the network. The Intrusion Detection system implemented is Anomaly-based, also RELU is used as an activation function.

Research made by (Parampottupadam et al.,2018) has provided a system to detect real-time Intrusion in cloud-based systems. They have used the NSL-KDD dataset to carry out the research. Although the efficiency is good, they have calculated the Results in terms of binary classification and Multiple classification.

Research made by (Diro et al., 2018) is primarily done for IoT-based infrastructure where a Distributed Intrusion Detection system is being established and accuracy is being captured in terms of Binary and Multiple classifiers, while NSL-KDD (UBMK'19) is being used as a dataset in it.

Research made by (Shenfield et al., 2018) uses a shellcode-based intrusion detection system, the Snort and Squil Data sets are used in it, and Deep learning is used with a Multilayer perceptron-based network architecture with 2 hidden layers in ANN.

In contrast, Aksu et al. (2018) proposed a deep learning architecture to detect port-scanning attacks and contrasted the results with the support vector machine (SVM) machine learning technique. CIC IDS 2017 dataset was used and there were 7 hidden layers in deep learning model, Also, RELU was the activation function. It will detect the breach if found through port number login patterns.

In the Research made by Kiflay et al. (2024), a multimodal-based NIDS framework is used where two RFC classifier models have been used, and the accuracy is being checked on the packet capture file of the UNSWNB-15 Dataset. Research made by (Wang et al., 2023) Implemented deep learning for IDS using Traditional Networks. Another Deep Learning classification framework has been proposed by the author (Aljuaid et al. 2024) where at the first stage feature selection will be done, the classification of the dataset will performed, using CNN as the base classifier, and analysis will be performed over cloud computing domain, and for feature selection and optimization, Pearson correlation coefficient will be selected.

Table 1. Prior work done in the field of IoT IDS, along with the limitations of each work

No.	Paper	Model Description/ Learning Algorithms	Used Datasets
1	(Lalduhsaka et al., 2022)	Anomaly-based IDS	UNB ISCX IDS 2012
2	(Attou et al. 2023)	A Cloud-Based Intrusion detection and prevention framework, Hybrid CNN-LSTM	IoT malware, Microsoft BIG-2015, and Malimg
3	(Diro et al., 2018)	Distributed IDS for IoT systems	NSL-KDD, 41 features
4	(Shenfield et al., 2018)	Shell code-based IDS	Snort and Sguil alert data
5	(Aksu et al., 2018)	Port Scanning Detection	CIC IDS 2017
6	(Kiflay et al., 2024)	Multimodal-based NIDS framework	UNSW-NB15
7	(Wang et al., 2023)	ResNet, Transformer and BiLSTM (Res-TranBiLSTM)	NSL-KDD dataset, CIC-IDS2017 dataset, and MQTTset dataset
8	(Aljuaid et al. 2024)	A Deep Learning Intrusion detection system for covering cyber-attacks in the cloud computing domain.	CSE-CICIDS2018
9	(Yang et al., 2017)	IDS for traditional networks.	NSL-KDD, 41 features
10	(Alazab et al., 2024)	HHO-MLP.	KDD Datasets
11	(Parampottupadam et al., 2018)	Real-time IDS for cloud systems.	NSL-KDD, 41 features
12	(Dawoud et al.,2018)	SDN-based IDS for IoT.	KDDCUP 99
13	(Grace et al. 2022)	A hybrid Intrusion detection framework has been proposed to identify malicious patterns. Hybrid LSTM-SVM.	CIC-AndMal-2017 datasets

The RNN-based intrusion detection system's structure was developed by the author (Yang et al., 2017) using the NSL-KDD dataset. Both binary and multiple classifications were used in this analysis. The test results for the RNN's learning rate selection and hidden node count were shown.

Also, the study done by Alazab et al. 2024) focuses on the model built using the deep learning-based approach, and the optimization technique used is the Harris Hawks Optimization algorithm (HHO). The MLP-HHO-based Intrusion Detection and Prevention System (IDS) works by adjusting weight as well as bias scenarios.

A cloud-based intrusion detection model has been presented by the authors (Attou et al. 2023). Due to the complex nature of the cloud environment, the classification is performed using a Random Forest classifier, for feature selection as well as an RF classifier, the model achieves good accuracy.

Research made by (Dawoud et al.,2018) defined a software-defined security solution prominently for IoT systems, where the dataset used is KDD CUP 99 proposed system that was a security solution based on software-defined networks for IoT systems.

The selection of features for an Internet-of-Things intrusion detection system (IDS)

The author of the paper (Roopak et al., 2020) proposed a multilayer IDS architecture based on Deep learning in which almost 26 features are selected using Matthew's correlation and obtained an accuracy of 98.3% for Denial of service attacks. In Fig 3 the accuracy percentage of multiple IDS Algorithms has been mentioned.

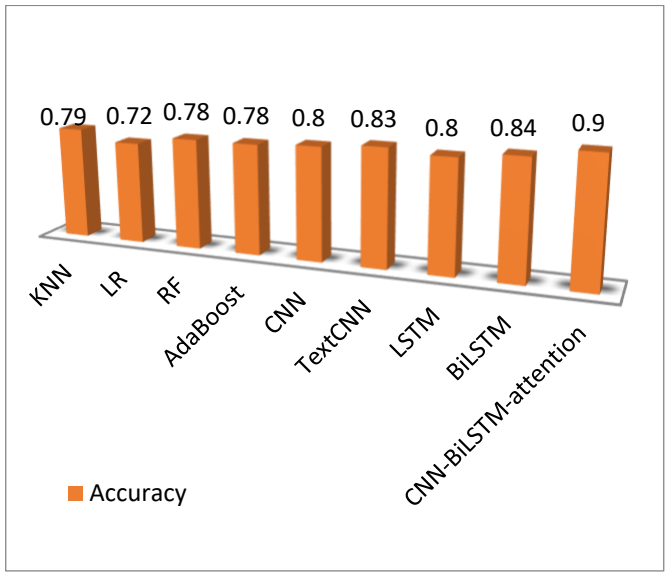


Fig. 3. Comparison of the accuracy of different IoT intrusion detection models.

In the study (Farhan et al. 2020), another Deep learning-based classifier (DNN) is being shown where data will be preprocessed using binary particle swarm optimization(BPSO), and then classification is done. Also as per the study given by author (Luke et al., 2024), we can observe the attacks done by intruders over multiple industrial sectors, as shown in Fig 4.

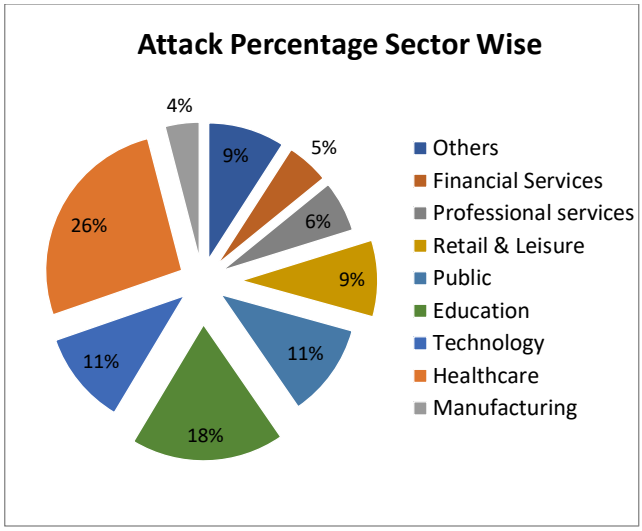


Fig. 4. Vulnerability percentage sector-wise (Luke et al., 2024)

In the study (Tsogbaatar et al. 2021) authors proposed a 'DeL-IoT' framework for intrusion detection, which works on the principle of Software-Defined Networking (SDN) phenomena. The meaningful features have been extracted using deep and stacked layered autoencoders. The proposed framework has shown a higher detection rate and accuracy in classifying malicious attack attempts, with 99.5- 99.9% in F-score, and achieved an accuracy of 91.04%-99.95% in MCC.

Research Gap

The dataset is significantly more important than any other module and forms the core of the model in machine learning-related systems. Selecting inaccurate or out-of-date datasets makes it useless for real-time deployment, especially in the information security space. Because of this, it will be challenging for any security expert to compile an up-to-date dataset that includes the latest attack scenarios and behavioral trends. Thus, one of the upcoming tasks could be upgrading the dataset. Also, when the literature studies are examined, it is observed that machine learning algorithms such as random forest, KNN, and XGBoost work well on some datasets; however, to perform better in the future, more hybrid models must be constructed by integrating algorithms for deep learning and machine learning on other upgraded datasets. LSTM, CatBoost, XGBM, RNN, and other machine-learning algorithms will also be tested. The major gap we identified is the lack of availability of enhanced and rich IoT datasets, also for the less efficient configured proposed models the small datasets are not available in IoT, and the datasets available are very huge, the IoT datasets should be made available in the ration of 10 % of the total dataset and 20 % of datasets accordingly due to large size of data. It will get very difficult to arrange the software and hardware requirements for such huge datasets for IoT and to apply the classifiers, so for the less capable proposed models (that cannot be directly applied to complete datasets) in terms of requirements a far smaller sample of IoT Datasets with the same capabilities should be proposed.

Proposed Model

The overall research will be performed in the following steps. The complete proposed algorithm flow chart can be seen in Fig. 6, and the proposed model in Fig. 8.

a) Dataset selection for the research.

b) Transformation and preprocessing of data.

c) Finding the scope of improvement.

d) Selecting the suitable ML Models for Evaluation over the Dataset.

e) Metric calculation and Evaluation with the metric of other ML Classifiers.

Proposed LEDCM Algorithm Description

The Above two Algorithms work in the combined view, It consists mainly of 2 Phases: first, to train the Model, and then to test the Model. LECDM tries to fetch the Leader Models using the following steps.

To find the Leader Models class-wise

i) *Training the Four Base Learners*:
In the Training Phase, we will train all four Base Learners, LightGBM, XGBoost, CatBoost, and RFC, using 80 percent of the Datasets.

ii) *Evaluation of Base Learners*:
The Four base Learners will be applied to test data, and their predictions, Accuracy, F1-scores, etc., will be evaluated on the datasets for each Category of Attacks. A detailed Analysis of base learners' predictions will be done.

iii) *Evaluate or Find the Leader Models or best-performing Models Class-wise*:
In this step, the best-performing models will be fetched as per their Accuracy % and F-scores class, the models that have the best values will be considered as Leader Model for the Class of Attack, if Two models have similar Accuracy % or scores, the models that are efficient and have less execution time, will be considered superior or Leader Model for that class.

iv) *Leader Model*:
The best-performing Base model for the class of Attack is considered the Leader Model.

4.1. Proposed Online ML Streaming LEDCM Algorithm: To Find The Leader Model

Pseudo code Algorithm 1: Leading Ensemble Decision Classifier Module (LEDCM)- Model Training

Data: Dtrain: The training dataset for the ML models
F = F1, F2, F3, F4 : the base ML models list used, F1= LightGBM Classifier, F2 = XGBoost Classifier, F3 = CatBoost Classifier, F4= Random Forest Classifier (RFC) C=1, 2, 3, 4.
n: to represent n different classes to which data entity can belong.
Result: F= F1, F2, F3, F4: the set of all considered trained base Models
LM = LF1, LF2,. ..... LFn: The SET of Leader Models Class wise to which data entity can belong
Step 1: Train all the considered Base Models.
F1→Training (F1, Dtrain); Represent to Train LightGBM Classifier using Dtrain data segment from original data.
F2→Training (F2, Dtrain); Represent to Train XGBoost Classifier using Dtrain data segment from original data.
F3→Training (F3, Dtrain); Represent to Train CatBoost Classifier using Dtrain data segment From original data.
F4→Training (F4, Dtrain); Represent to Train Random forest Classifier using Dtrain data segment from original data.
Step 2: To fetch the leader model framework individually for every data label (s) (normal or any type of attack)
initialization;
for *iteration* = 1, 2, ...*n* .....**do**
  Flistc → Bestperforming
  (F1, F2, F3, F4, C: used to show the class of label or type of attack) The framework with the highest f1 scores will be considered the best-performing model for the corresponding Attack of class category. (i.e. class can represent normal or any type of attack). ;
  if *Len (Flistc) = = 1 // The above condition represents that there exists only one framework as a leader model for Specific label type, so we will consider it to be the leader model for that class label*
  then
    LFC → FListC[0]; ;
    // It represents, saving the current framework or model as a leader model for the specific Class of label type Ci.
  else
    // when more than 1 ML Model has a similar F1-score for the class type LFC → MostEfficient(FListC);
    //saving the quickest reliable Model as the Superior framework for class or label type C.
  end
  LF → LF U LFC: // represent to save the Superior Model or framework for each class labels
end

Fig. 5. Proposed Online ML Streaming LEDCM Algorithm

LEDCM Prediction Mechanism

i) *Initial Predictions*:

For Every Tuple, all four Models will be evaluated, and their predictions will be compared and stored for concurrent Evaluation**.**

ii) *All four models have the same predicted class, Same Predictions***:**

If the predicted classes by all four Models are equal, then anyone can be qualified to be the predicted model**.**

iii) *All four models have different predicted Classes*:

(a) If the prediction made by the base learner is the same as that by the Leader model, and they are the

same by nature (i.e., such as RFC is the base model and RFC itself is the Leader model for that class of attack), then that particular base model can be considered.

(b) If more than two base models and Leader Models pairs exist, then the one with the Highest confidence will be considered for prediction.

(c) If all the classes have different predictions for an Individual tuple, and none of the Base and Leader Model pair predictions match, then the one (i.e., Base Model) having the highest confidence percentage will be considered as the predicted Model if none match with the Leader Model.

(d) Further below, for the same scenarios, a flowchart has been drawn for a better understanding of all the multiple possible scenarios.

More than 2 Models have the same predicted class, and others have different ones:

If the MODE of value of the similar prediction counts by the base models is greater than 2 and less than 4, then the base learner will be considered based on the Leader model comparison; whichever is the Leader model in that case will be considered as the prediction Model. IoT networks generate a massive amount of data, which requires preprocessing to remove noise, handle missing values, and normalize the data. Techniques such as data cleaning, feature scaling, and feature engineering are applied to improve the data format in terms of values. This step is one of the most important steps that helps the machine learning model to learn and evaluate in a better way, by removing the unnecessary features that do not have significance in the overall evaluation. Many feature selection algorithms are available that can be deployed. Dimension reduction algorithm is better utilized in the case when the datasets have huge feature complexity, thus, we first try to filter it out, the base of datasets, features, and ML models, as a suitable feature selection method can be used.

Data Collection and Processing

The common IoT dataset, which accurately reflects modern-day traffic, is used in this work (NSL-KDD, UNSW-NB15, NF-BOT-IOT datasets), shown in Table 2. By utilizing the collective wisdom of multiple decision trees, Random Forest's feature selection process helps identify the most influential features in a dataset, contributing to better model accuracy and interpretability. Features with higher importance are considered more relevant for prediction and are retained, while features with lower importance can be dropped (Table 3).

There are multiple authentic sources from which we can fetch the datasets. The dataset contains several intrusions that were simulated in an intelligence network context, The raw TCP/IP Network traffic data will be fetched observed simulated models will be created and multiple known attacks will be bombarded on the hosts of networks in the simulated environment and all the normal and abnormal parameters will be saved in some CSV or excel or notepad files(i.e. these files will further be considered as the Raw datasets for IDS scenarios) and attributes will be stored for few days, hours, etc. The simulated environments will have the two types of data according to their behavior.

a) Normal

b) Abnormal

Also, one of the problems faced in the Available datasets for network analysis or IoT analysis is that it does not contain the proper distribution of class labels for attacks in terms of tuples in CSV files or text files, etc., like there is uneven distribution of data among various class labels, so we could not get the good strength for Model train and testing purpose and faces issues. If we want the classifier should have good prediction capability and accuracy for the class attack labels then we cannot compromise in terms of the provided records in the datasets, because for any specific class if we have fewer records or tuples then we will face training issues in terms of performance, if the classifier will be trained on lesser data for class label and expectations are high, the Artificial Neural Networks (ANNs) will be trained on high-dimensional data received from previous stages of fraudulent detection to predict whether there is any possibility of data or information breach in the existing systems, and if any, the system will automatically stop all the data or information flows without going through any losses. In the current study, we will discuss intrusion detection systems implemented through Deep Learning algorithms (DL), also about the parameters on which DL procedures work, and how they will prevent Cyber intelligence systems (CIS) from getting security breached.

Networks generate a massive amount of data, which requires preprocessing to remove noise, handle missing values, and normalize the data. Techniques such as data cleaning, feature scaling, and feature engineering are applied to improve the data format in terms of values. Feature selection is one of the most important steps that helps the machine learning model to learn and evaluate in a better way, by removing the unnecessary features that do not have significance in the overall evaluation. Many feature selection algorithms are available that can be deployed. Dimension reduction algorithm is better utilized in the case when the datasets have huge feature complexity, thus, we first try to remove irrelevant features.

The benefit we have of using a hybrid framework of machine learning classifiers for malicious intent detection in cyber security is their ability to gain or increase the overall performance, accuracy, and robustness of the framework, with the combination of different classifiers, the hybrid framework can better generalize patterns and adapt to evolving attack techniques. In the case of unbalanced datasets where we see huge variations in the number of records between multiple class labels (Attack types), hybrid frameworks will outperform the capabilities of individual classifiers.

There are multiple authentic sources from which we can fetch the datasets. The dataset contains several intrusions that were simulated in an intelligence network context, The raw TCP/IP Network traffic data will be fetched observed simulated models will be created and multiple known attacks will be bombarded on the hosts of networks in the simulated environment and all the normal and abnormal parameters will be saved in some CSV or excel or notepad files(i.e. these files will further be considered as the Raw datasets for IDS scenarios) and attributes will be stored for few days, hours, etc.

The simulated environments will have two types of data according to their behavior, normal and abnormal. If the classifier is trained using the unbalanced data labels, then there is a possibility of getting variation in classifier precision capability. Some specific class labels have to underfit or overfit data records, and so the predictions will be made accordingly.

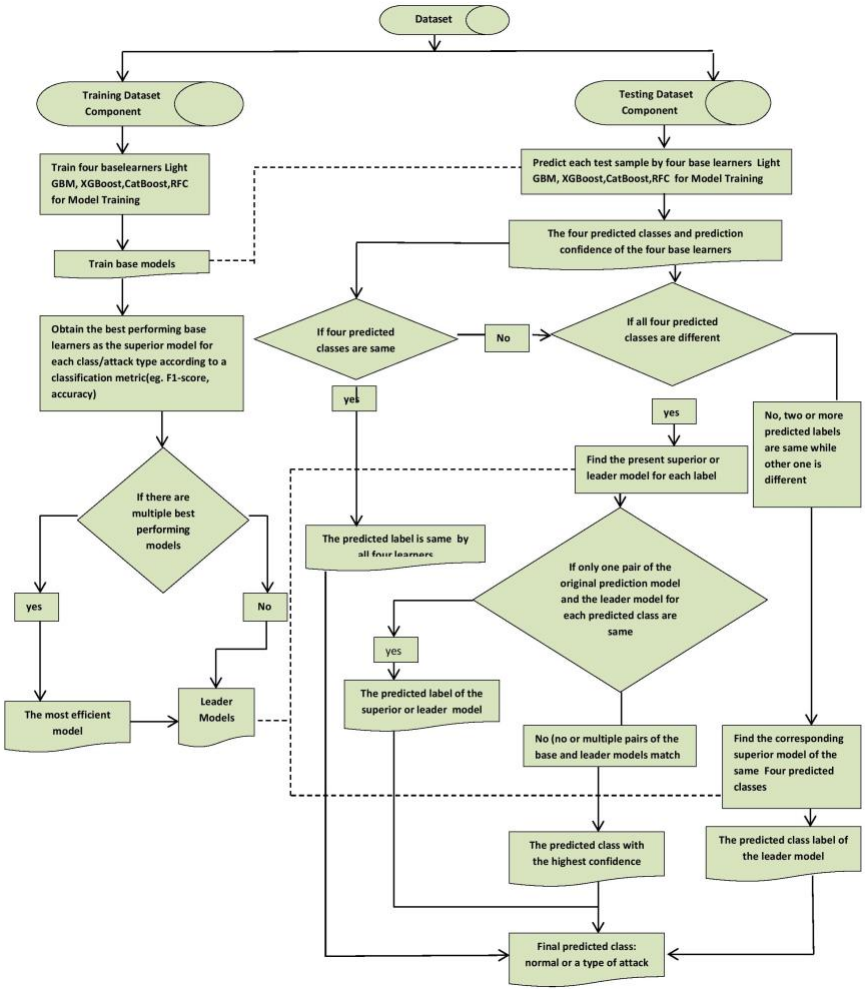


Fig. 6. Flow chart

Pseudo code Algorithm 2 : Leading Ensemble Decision Classifier Module (LEDCM) streaming Algorithm- Model Prediction

Data: D_{test}: The testing dataset for the ML models.(Applying, All Four Base Models Simultaneously on Data iteratively), **Leader Model class wise from** Algorithm-1.
F = $\{F_1, F_2, F_3, F_4\}$: the base ML models list used, F_1= LightGBM Classifier, F_2 = XGBoost Classifier, F_3 = CatBoost Classifier, F_4= Random Forest Classifier (RFC)}
C=1, 2, 3, 4…… **n**: to represent *n* different classes to which data entity can belong.
Result: Ltest: predicted accurate classes to which test sample elements belong.

for each data record sample$x_i \in D_{test}$ **do** //for each single test temple
  $K_{i1}, p_{i1} \leftarrow Prediction\ (F_1, x_i)$; // used to represent the prediction made by trained LightGBM Classifier on test samples also to save the confidence probability and the class label to which a record belongs.
  $K_{i2}, p_{i2} \leftarrow Prediction\ (F2, x_i)$; // prediction on D_{test} by using XGBoost Classifier
  $K_{i3}, p_{i3} \leftarrow Prediction\ (F3, x_i)$; // prediction on D_{test} by using CatBoost Classifier
  $K_{i4}, p_{i4} \leftarrow Prediction\ (F4, x_i)$; // prediction on D_{test} by using RFC Classifier
  if $(K_{i1} == K_{i2} == K_{i3} == K_{i4})$ **then** //if all the four frameworks or models have the same prediction label for data record D_{test}
    $K_i \leftarrow K_{i1}$ //is used to represent that the predicted label is a final predicted class
  else
    if $(K_{i1} != K_{i2} != K_{i3} != K_{i4})$ **then: :**// if the predicted class is different by all the four models
      for $j = 1,2,3,4$ ***do*** // for each classification framework perform the below check
        if $(F_j == KF_{Li,j})$ **then:** //Finding the classified label of the original framework is similar to leader framework obtained in **Algorithm-1**
          $K\_list_i \leftarrow K\_list_i \cup \{K_{i,j}\}$; //for each iteration step saving or keeping the predicted label
          $q\_list_i \leftarrow q\_list_i \cup \{p_{i,j}\}$; //for each iteration step save the predicted class confidence
        end
      end
      else
        if Len$(K_list_i) == 1$ **then: //** This condition shows if only a single pair of the original classifier Framework and superior model framework exists that is equal for the class label
          $L_j \leftarrow K\_list_i[0]$ // uses the classified label class for data record of the leader Model framework from *algorithm 1 as predicted class;*
        else*// used to show if none pair or many pairs of original classifier and the superior model for classified label for record exists*
        if Len$(K_list_i) == 0$ **then:**
          $q\_list_i \leftarrow \{p_{i1}, p_{i2}, p_{i3}, p_{i4}\}$; // Used to exempt void probability list
        end
        $q\_max_i \leftarrow \max(q\_list_i)$ // to find the maximum value of confidence
        if $(q\_max_i == p_{i1})$ **then:** // using to show the highest percentage label class with a maximum as the final prediction class
          $K_i \leftarrow K_{i1}$; // step will be executed when LightGBM has the highest confidence
        elseif $(p\_max_i == p_{i2})$ **then**:
          $K_i \leftarrow K_{i2}$; // step will be executed when XGBoost has the highest confidence
        Elseif $(p\_max_i == p_{i3})$ **then**:
          $K_i \leftarrow K_{i3}$; // step will be executed when CatBoost has the highest confidence
        else
          $K_i \leftarrow K_{i4}$;// step will be executed when RFC has the highest confidence
        end
      end
    else
      // condition to represent when two or more classifiers have the same predicted class label for a record
      n ← **mode** $(K_{i1}, K_{i2}, K_{i3}, K_{i4},)$ // fetching the predicted class label with the majority number of votes.
      $K_i Prediction(F_n, x_i)$;// Represent the case when leader model prediction of the class label for data has been considered as the final prediction class
  end
  $K_{test} \leftarrow K_{test} \cup \{K_i\}$; // Saving the predicted class labels for all tested data record samples;
end.

Fig. 7. Pseudo code Algorithm 2

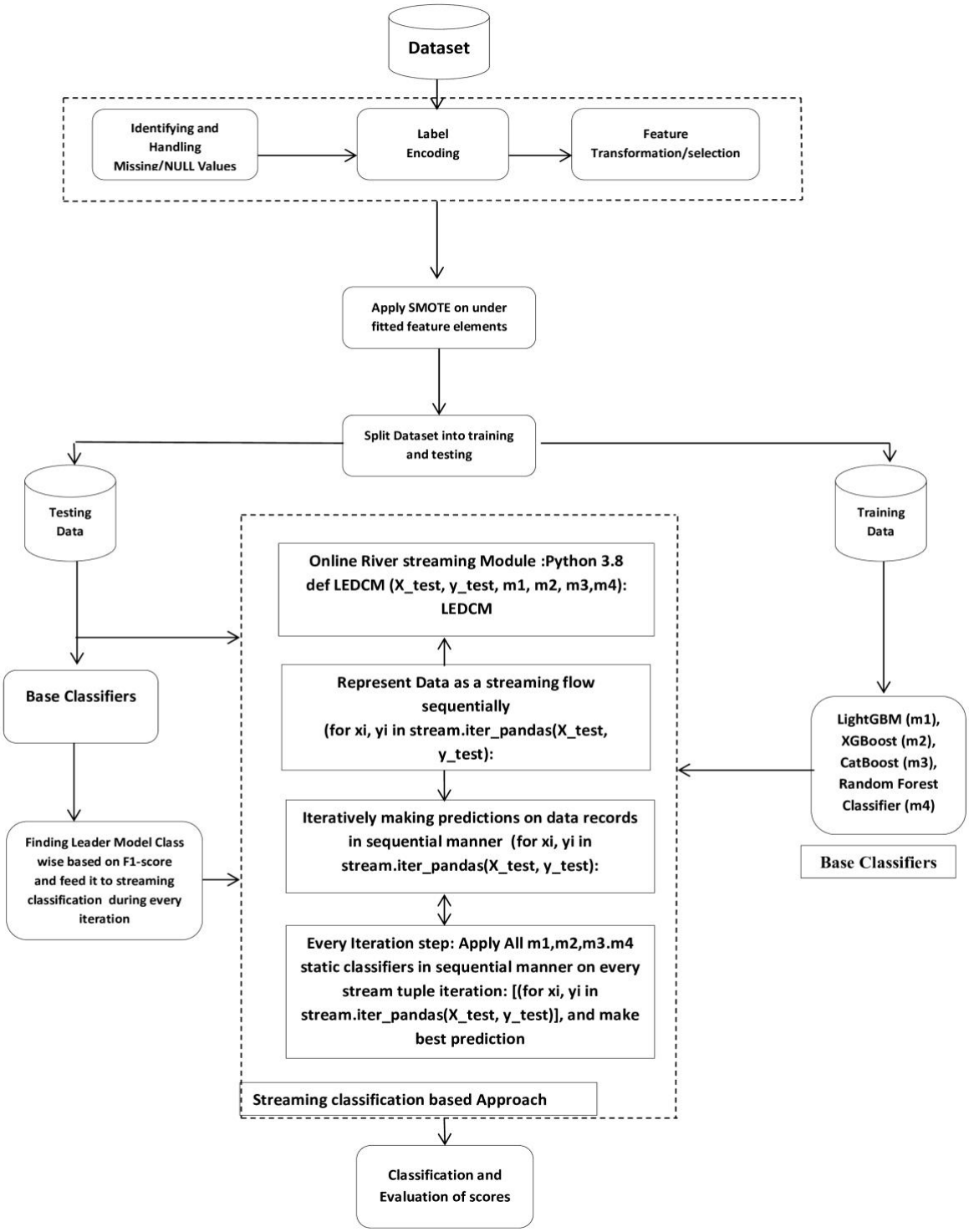


Fig. 8. Proposed Model Online ML streaming, LEDCM

Table 2. Datasets Used in the Current Scenario

Used Dataset	File specified	Label Types	Training Size	Testing Size
UNSW_NB15	UNSW_NB15	Normal, Backdoors, DoS, Exploits, Generic, Worms, Reconnaissance, Fuzzers, Analysis.	64938	16235
NSL-KDD	KDD cup.data_10_percent	Normal,DoS,R2L,U2R, Probe	118,835	29,704
NF-BOT-IOT	NF-BOT-IOT-v1	DDoS, DoS, Benign, Theft, Reconnaissance	480080	120020

Table 3. Datasets Features Used

Dataset	Attributes used
UNSW_NB15	dur, proto, service, state,spkts,dpkts,sbytes,dbytes, rate,sttl,dttl,sload,dload, sloss,dloss,sinpkt,dinpkt,sjit,djit, swin, stcpb, dtcpb, dwin, tcprtt, synack, ackdat, smean, dmean, trans_depth, response_body_len, ct_srv_src, ct_state_ttl, ct_dst_ltm, ct_src_dport_ltm, ct_dst_sport_ltm, ct_dst_src_ltm, is_ftp_login, ct_ftp_cmd, ct_flw_http_mthd, ct_src_ltm, ct_srv_dst,is_sm_ips_ports
NSL-KDD	duration,protocol_type,service,flag,src_bytes,dst_bytes,land,wrong_fragment,urgent,hot,num_failed_logins,logged_in,num_compromised,root_shell,su_attempted,num_root,num_file_creations,num_shells,num_access_files,num_outbound_cmds,is_host_login,is_guest_login,count,srv_count,serror_rate,srv_serror_rate,rerror_rate,srv_rerror_rate,same_srv_rate,diff_srv_rate,srv_diff_host_rate, dst_host_count,dst_host_srv_count,dst_host_same_srv_rate,dst_host_diff_srv_rate,dst_host_same_src_port_rate,dst_host_srv_diff_host_rate,dst_host_serror_rate,dst_host_srv_serror_rate,dst_host_rerror_rate,dst_host_srv_rerror_rate,label
NF-BOT-IOT	IPV4_SRC_ADDR, L4_SRC_PORT, IPV4_DST_ADDR, L4_DST_PORT, PROTOCOL, FLOW_DURATION_MILLISECONDS, L7_PROTO,IN_BYTES,OUT_BYTES,IN_PKTS,OUT_PKTS,TCP_FLAGS Label, Attack

It is not certain that it will perform well and give good precision and F1 scores. Below are the major limitations observed while considering the datasets and then training the classifier.

a) There is an uneven distribution of tuples and records class or label-wise (i.e., in other words, we can say in the overall datasets the number of records for Attack label A is less than the Attack label B, and the same hierarchy we will be seeing in the whole dataset). It is a common example of uneven distribution of labels in terms of records.

b) If the classifier is trained using the unbalanced data labels, then there is a possibility of getting variation in classifier precision capability.

c) Some specific class labels have to be fitted to the data records, and so the predictions will be made.

d) Some specific class labels have overfit data records, and so the predictions will be made.

The solution for the above issue will be covered by the SMOTE Algorithm.

Synthetic Minority Oversampling Technique (SMOTE)

This technique works in many ways for the Input datasets and helps to remove the class label imbalance ratios, We can fit the SMOTE function as per our standing or need. Either we can directly apply the SMOTE algorithm function on the complete dataset, or it will automatically adjust the data records class-wise, either by replication or more generation of records for under fit class label. We can also specify the value in terms of an integer up to which we want the class label records to be scaled up or increased.

Below are the two common modes of operation that SMOTE provides

a) To directly call the smote fit function in a way that the algorithm or procedure will automatically adjust or scale the underfit class records.
b) To provide the threshold value up to which any specific class label records we want to replicate or enlarge.

After applying the SMOTE we can expect to see an increase in the prediction capabilities of the data, and the classifiers will be trained on the equal distribution of data records as per class labels, SMOTE is a very rich classifier efficiency technique and can be adopted for the scenarios of datasets having uneven class label distribution in data files.

Materials and Methods

We have implemented proposed LEDCM Algorithm using Google Colab platform, the implementation is done on python and the version used is Python 3.10, with runtime environment as python 3 and multiple Machine Learning libraries such as River, numpy, pandas etc. Hardware accelerator is used as CPU and T4 GPU cycles for execution.

Result Analysis and Discussion

Evaluation metrics

To validate outcomes, various efficiency appraisals, including FP, FN, TN, and TP, are used. These evaluation metrics are used to measure statistics by different academics to get the results.

a) Confusion Matrix (CM):
The confusion matrix is used to find the correlation between the actual and expected classes. It is also thought to be a helpful statistic for estimating AUC and ROC curves, specificity, precision, recall, and accuracy. Table 4 displays the confusion matrix.

Table 4. Confusion Matrix (CM)

Class	Actual Positive Class	Actual Negative Class
Predicted Positive Class	TP	FP
Predicted Negative Class	FN	TN

b) Accuracy:
The percentage of instances that are correctly classified is calculated by:

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

c) Error Rate:
The percentage of predicted values that are incorrectly categorized is determined by

$$\text{Error rate} = 1 - \text{Accuracy}$$

d) True Positive Rate :
Utilizing this metric allows for the accurate measurement of actual positive proportions. The TPR is acquired using.

$$\text{True positive rate} = \frac{TP}{TP+FN}$$

e) False positive rate:
If it is true, the null hypothesis is rejected. The FPR can be found by:

$$\text{False positive rate} = \frac{FP}{FP+TN}$$

f) True negative rate:
The regular instances of the patterns are precisely identified. The TNR was discovered by:

$$\text{True negative rate} = 1 - FPR$$

g) False Negative Rate:
The patterns are incorrectly categorized as common occurrences. The FNR can be acquired by:

$$\text{False Negative rate} = 1 - TPR$$

h) Precision:
The patterns are accurately classified by the extent of their action. It is obtained by:

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

i) F-Measure:
The appraisal of the accuracy is performed by it. It is obtained by:

$$\text{F-measure } 2 \times \frac{\text{precision x recall}}{\text{precision}+\text{recall}}$$

Result Analysis

This study combines the results and metrics fetched from 4 base classifiers LightGBM, Random forest classifier (RFC), XGBoost, and CatBoost, and merged it into a Combined Leading Ensemble Decision Classifier Module (LEDCM) for the prediction of every record or tuple to find the accurate class it belongs to in case of Multiclass classification scenarios on IoT based Datasets (i.e. NSL-KDD, NF-BOT-IOT, UNSW-NB-15 datasets), LEDCM merges the prediction from all the major 4 classifiers in such a way, the overall Accuracy % and f1-score will get increased by fetching the best prediction from all base classifiers in a group individually Class wise. In the final results, we observed that our proposed model got the highest accuracy of 99.94% for NSL-KDD test data, 95.99% for UNSW-NB-15, and 83.13% for NF-BOT-IOT for Multiclass classification (from Table 8-10), and Accuracy of 100% for all three Datasets in case of

binary label classification in comparison to other base classifiers (from table 5-7). The confusion matrices for both multiclass and binary classification are shown in Fig. 6 through Fig. 11.

Table 5. NSL-KDD performance matrix for binary label classification

Classifier	Data type	Precision	Recall	F1 Score	Accuracy (%)
Light GBM	Simple/Normal	1.00	1.00	1.00	100.00
	Breach	1.00	1.00	1.00	
XGBoost	Simple/Normal	1.00	1.00	1.00	100.00
	Breach	1.00	1.00	1.00	
CatBoost	Simple/Normal	1.00	1.00	1.00	100.00
	Breach	1.00	1.00	1.00	
Random Forest Classifier	Simple/Normal	1.00	1.00	1.00	100.00
	Breach	1.00	1.00	1.00	
Proposed, LEDCM model	Simple/Normal	1.00	1.00	1.00	100.00
	Breach	1.00	1.00	1.00	

Table 6. UNSW-15 performance matrix for binary label classification

Classifier	Data type	Precision	Recall	F1 score	Accuracy (%)
Light GBM	Simple/Normal	1.00	1.00	1.00	100.00
	Breach	1.00	1.00	1.00	
XGBoost	Simple/Normal	1.00	1.00	1.00	100.00
	Breach	1.00	1.00	1.00	
CatBoost	Simple/Normal	1.00	1.00	1.00	100.00
	Breach	1.00	1.00	1.00	
Random Forest Classifier	Simple/Normal	1.00	0.998	1.00	99.85
	Breach	1.00	0.998	1.00	
Proposed, LEDCM model	Simple/Normal	1.00	1.00	1.00	100.00
	Breach	1.00	1.00	1.00	

Table 7. NF-BOT-IOT performance matrix for binary label classification

Classifier	Data type	Precision	Recall	F1 score	Accuracy (%)
Light GBM	Simple/Normal	1.00	1.00	1.00	100.00
	Breach	1.00	1.00	1.00	
XGBoost	Simple/Normal	1.00	1.00	1.00	100.00
	Breach	1.00	1.00	1.00	
CatBoost	Simple/Normal	1.00	1.00	1.00	100.00
	Breach	1.00	1.00	1.00	
Random Forest Classifier	Simple/Normal	1.00	1.00	1.00	100.00
	Breach	1.00	1.00	1.00	
Proposed LEDCM model	Simple/Normal	1.00	1.00	1.00	100.00
	Breach	1.00	1.00	1.00	

Table 8. KDD'99 performance matrix result (multiclass classification)

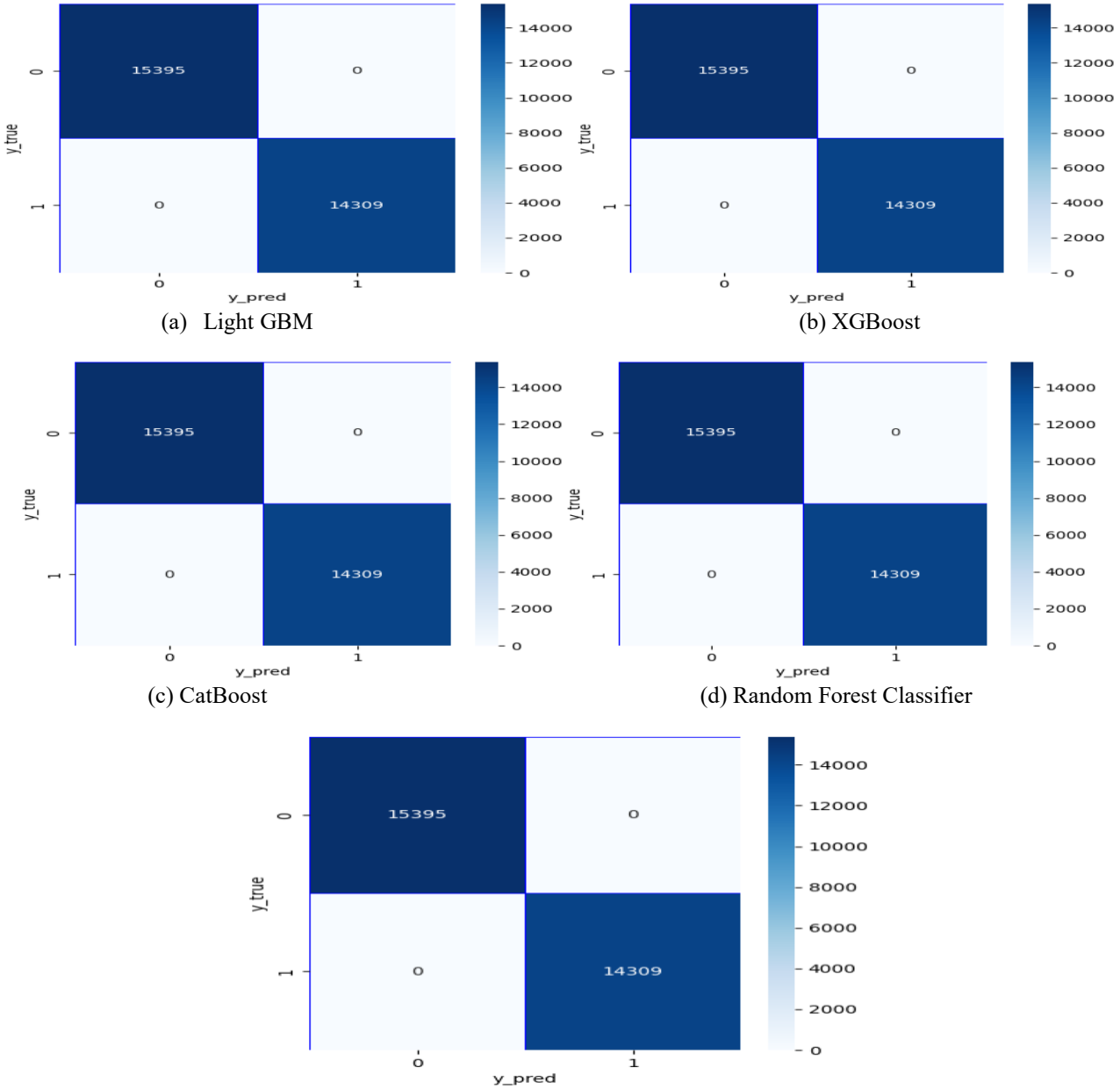
Classifier	Data type	Precision	Recall	F1-score	Accuracy (%)
Light GBM	Normal	1.00	1.00	1.00	99.93
	DoS	1.00	1.00	1.00	
	U2R	0.78	0.88	0.82	
	R2L	0.92	0.96	0.94	
	Probe	1.0	1.00	1.00	
XGBoost	Normal	1.00	1.00	1.00	99.92
	DoS	1.00	1.00	1.00	
	U2R	0.88	0.92	0.90	
	R2L	1.00	0.99	0.99	
	Probe	1.00	1.00	1.00	
CatBoost	Normal	1.00	1.00	1.00	99.89
	DoS	1.00	1.00	1.00	
	U2R	0.85	0.92	0.80	
	R2L	0.99	0.99	0.99	
	Probe	1.00	1.00	1.00	
RFC Classifier	Normal	1.00	1.00	1.00	99.90
	DoS	1.00	1.00	1.00	
	U2R	0.92	0.96	0.94	
	R2L	0.99	0.99	0.99	
	Probe	1.00	1.00	1.00	
Proposed model	Normal	1.00	1.00	1.00	99.94
	DoS	1.00	1.00	1.00	
	U2R	0.92	0.96	0.94	
	R2L	1.00	0.99	0.99	
	Probe	1.00	1.00	1.00	

Table 9. UNSW–NB15 class label response (multiclass classification)

Classifier	Data type	Precision	Recall	F1-score	Accuracy (%)
Light GBM	Normal	1.00	1.00	1.00	95.70
	Backdoor	0.55	0.27	0.36	
	Analysis	0.78	0.89	0.83	
	Fuzzers	0.96	0.95	0.96	
	Exploits	0.88	0.94	0.91	
	Reconnaissance	0.57	0.61	0.59	
	DoS	0.65	0.30	0.41	
	Worms	0.65	0.56	0.60	
	Generic	1.00	1.00	1.00	
XGBoost	Normal	1.00	1.00	1.00	95.58
	Backdoor	0.58	0.32	0.41	
	Analysis	0.76	0.87	0.81	
	Fuzzers	0.97	0.94	0.95	
	Exploits	0.88	0.94	0.91	
	Reconnaissance	0.57	0.57	0.57	
	DoS	0.64	0.29	0.40	
	Worms	0.57	0.48	0.52	
	Generic	1.00	1.00	1.00	
CatBoost	Normal	1.00	1.00	1.00	95.87
	Backdoor	0.50	0.27	0.35	
	Analysis	0.67	0.88	0.76	
	Fuzzers	0.96	0.93	0.94	
	Exploits	0.88	0.94	0.91	
	Reconnaissance	0.57	0.57	0.57	
	DoS	0.61	0.25	0.36	
	Worms	0.57	0.48	0.52	
	Generic	1.00	1.00	1.00	
RFC Classifier	Normal	1.00	1.00	1.00	95.84
	Backdoor	0.45	0.41	0.43	
	Analysis	0.72	0.85	0.78	
	Fuzzers	0.96	0.93	0.95	
	Exploits	0.89	.93	0.91	
	Reconnaissance	0.60	0.61	0.60	
	DoS	0.61	0.36	0.45	
	Worms	0.64	0.67	0.65	
	Generic	1.00	0.99	1.00	
Proposed model	Normal	1.00	1.00	1.00	95.99
	Backdoor	0.44	0.36	0.40	
	Analysis	0.73	0.88	0.80	
	Fuzzers	0.95	0.95	0.95	
	Exploits	0.89	0.93	0.91	
	Reconnaissance	0.60	0.61	0.60	
	DoS	0.61	0.36	0.45	
	Worms	0.67	0.67	0.67	
	Generic	1.00	1.00	1.00	

Table 10. NF-BOT-IOT class label response (multiclass classification)

Classifier	Data type	Precision	Recall	F1-score	Accuracy (%)
Light GBM	DDoS	0.36	0.23	0.28	81.86
	DoS	0.35	0.58	0.43	
	Benign	0.50	0.33	0.40	
	Theft	0.02	0.02	0.02	
	Reconnaissance	0.96	0.94	0.95	
XGBoost	DDoS	0.31	0.32	0.31	82.71
	DoS	0.29	0.28	0.29	
	Benign	1.0	1.0	1.0	
	Theft	0.84	0.83	0.84	
	Reconnaissance	0.95	0.95	0.95	
CatBoost	DDoS	0.31	0.31	0.31	82.88
	DoS	0.30	0.28	0.29	
	Benign	1.0	1.0	1.0	
	Theft	0.86	0.84	0.85	
	Reconnaissance	0.95	0.95	0.95	
RFC Classifier	DDoS	0.05	0.05	0.05	76.90
	DoS	0.05	0.05	0.05	
	Benign	1.0	1.0	1.0	
	Theft	0.82	0.83	0.83	
	Reconnaissance	0.93	0.93	0.93	
Proposed model	DDoS	0.29	0.27	0.28	83.13
	DoS	0.29	0.38	0.33	
	Benign	1.0	1.0	1.0	
	Theft	0.56	0.84	0.67	
	Reconnaissance	0.97	0.93	0.95	

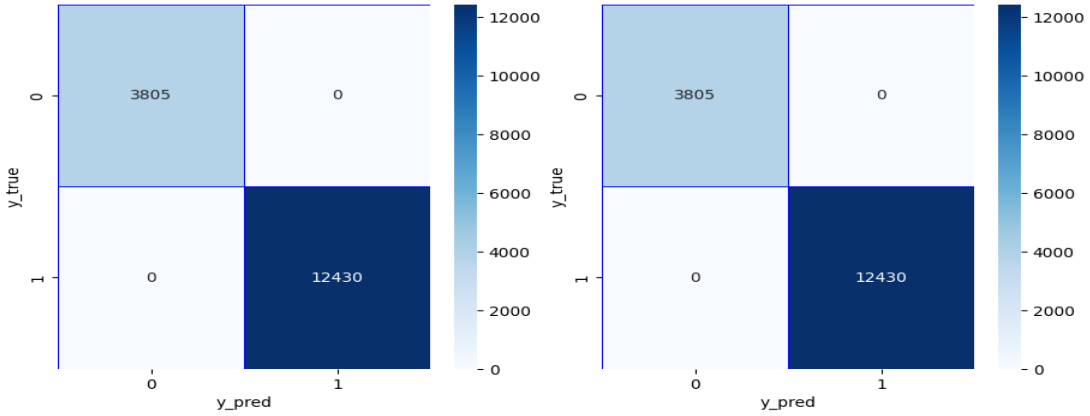


(a) Light GBM (b) XGBoost

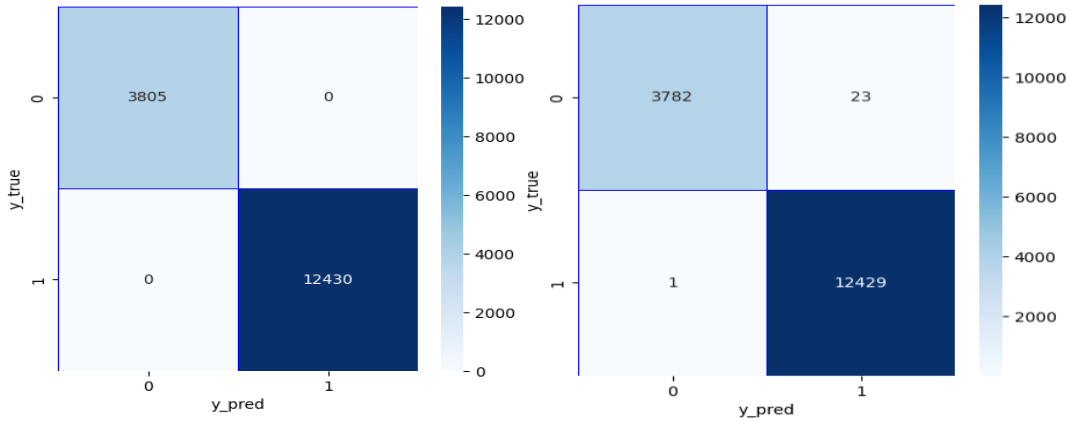
(c) CatBoost (d) Random Forest Classifier

(e)Proposed LEDCM model

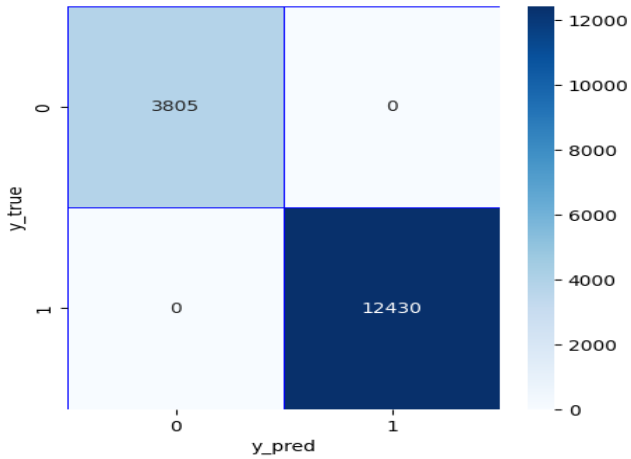
Fig. 9. NSL-KDD confusion matrix (Binary classification)



(a)Light GBM (b) XGBoost

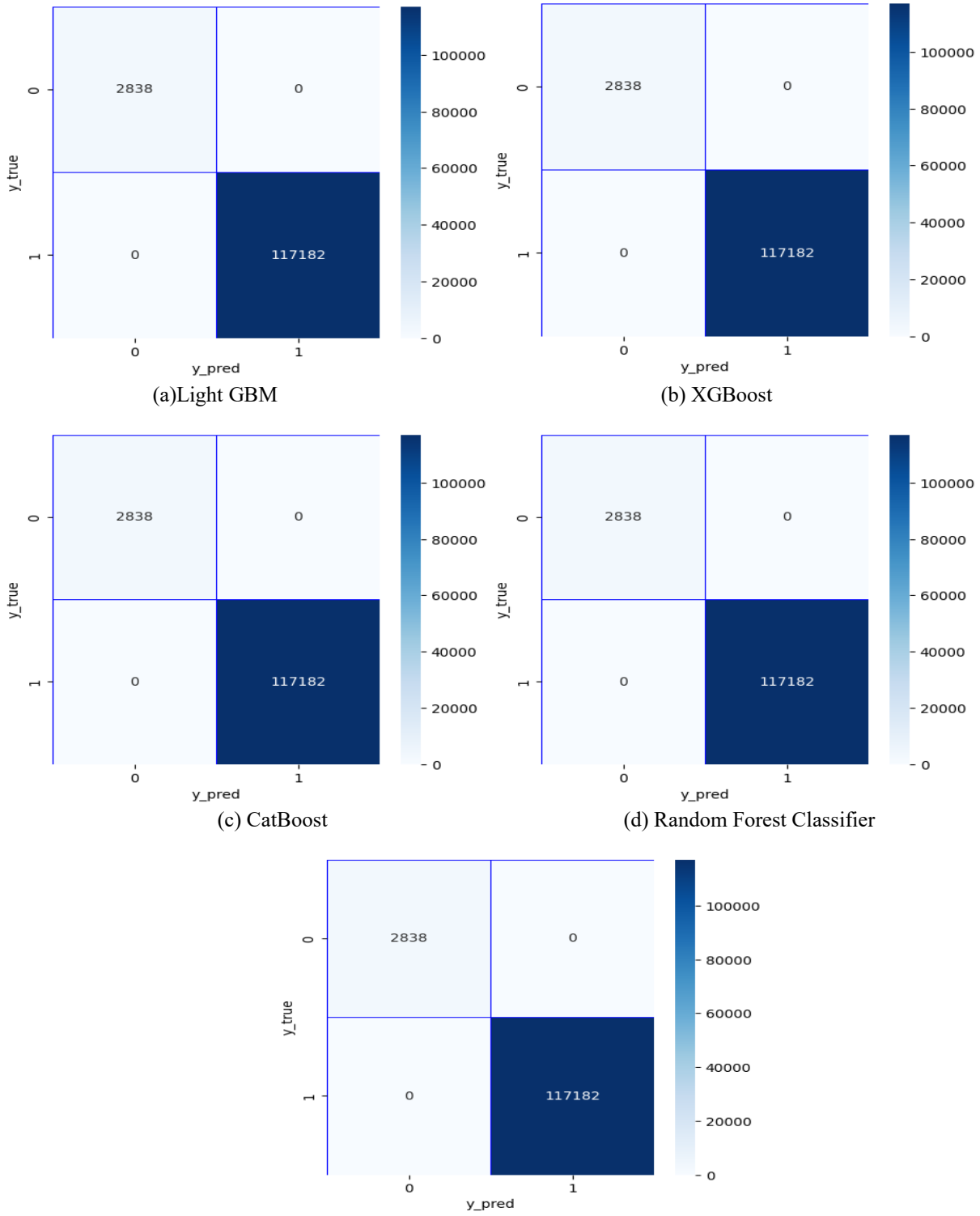


(c) CatBoost (d) Random Forest Classifier



(e)Proposed LEDCM model

Fig. 10. UNSW-NB15 confusion matrix (Binary classification)

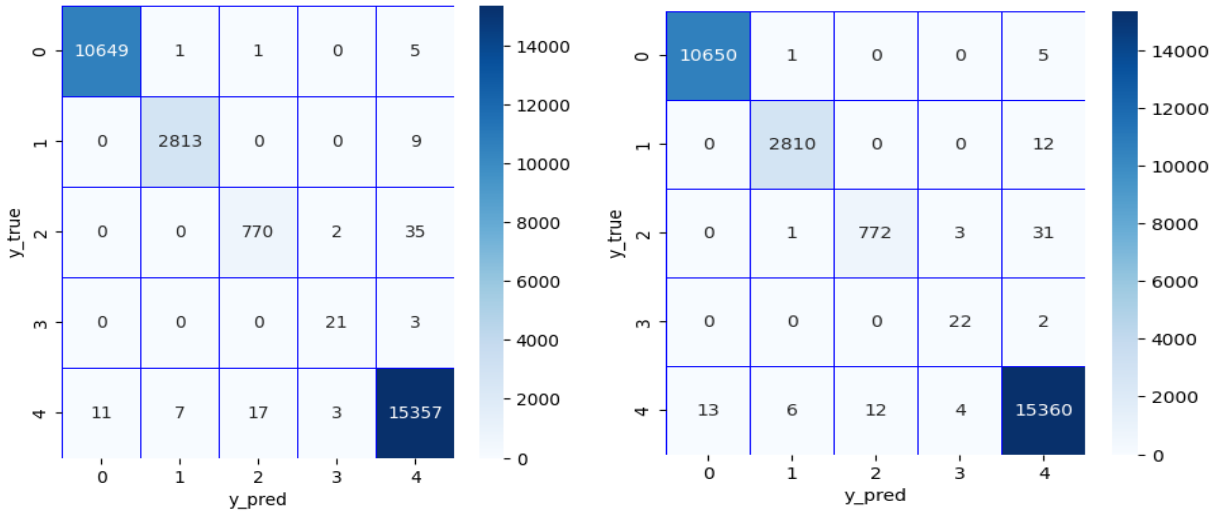


(a)Light GBM (b) XGBoost

(c) CatBoost (d) Random Forest Classifier

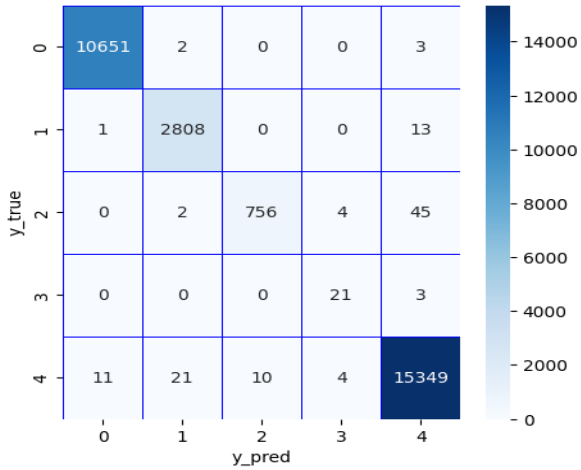
(e)Proposed LEDCM model

Fig. 11. NF-BOT confusion matrix (Binary classification)

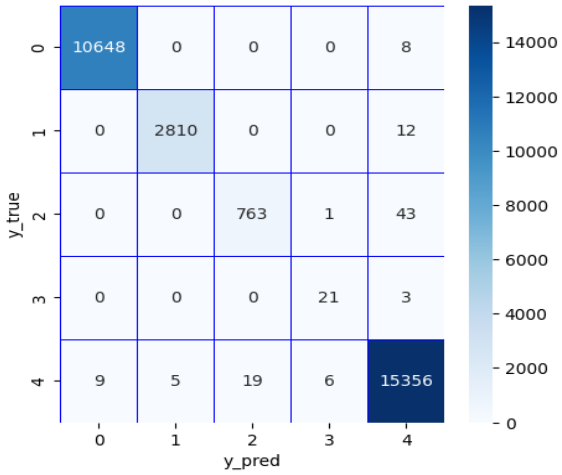


(a)Light GBM

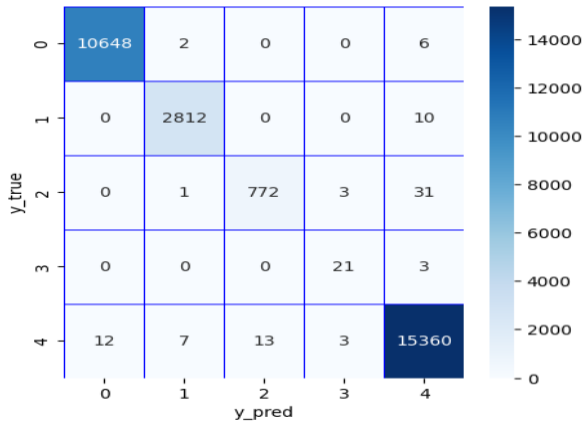
(b) XGBoost



(c) CatBoost

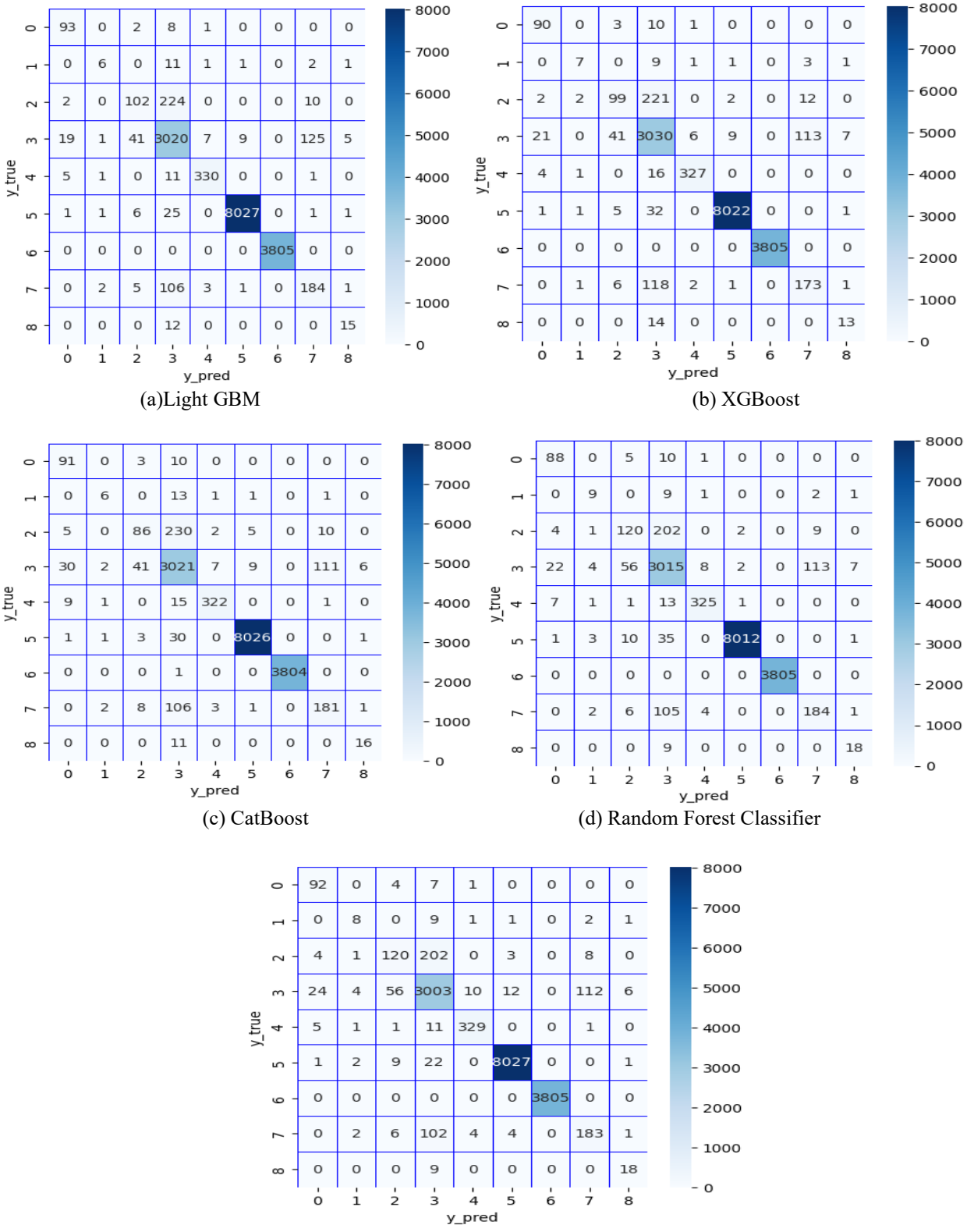


(d) Random Forest Classifier



(e) Proposed LEDCM model

Fig. 12. NSL-KDD confusion matrix (Multiclass classification)



(a)Light GBM

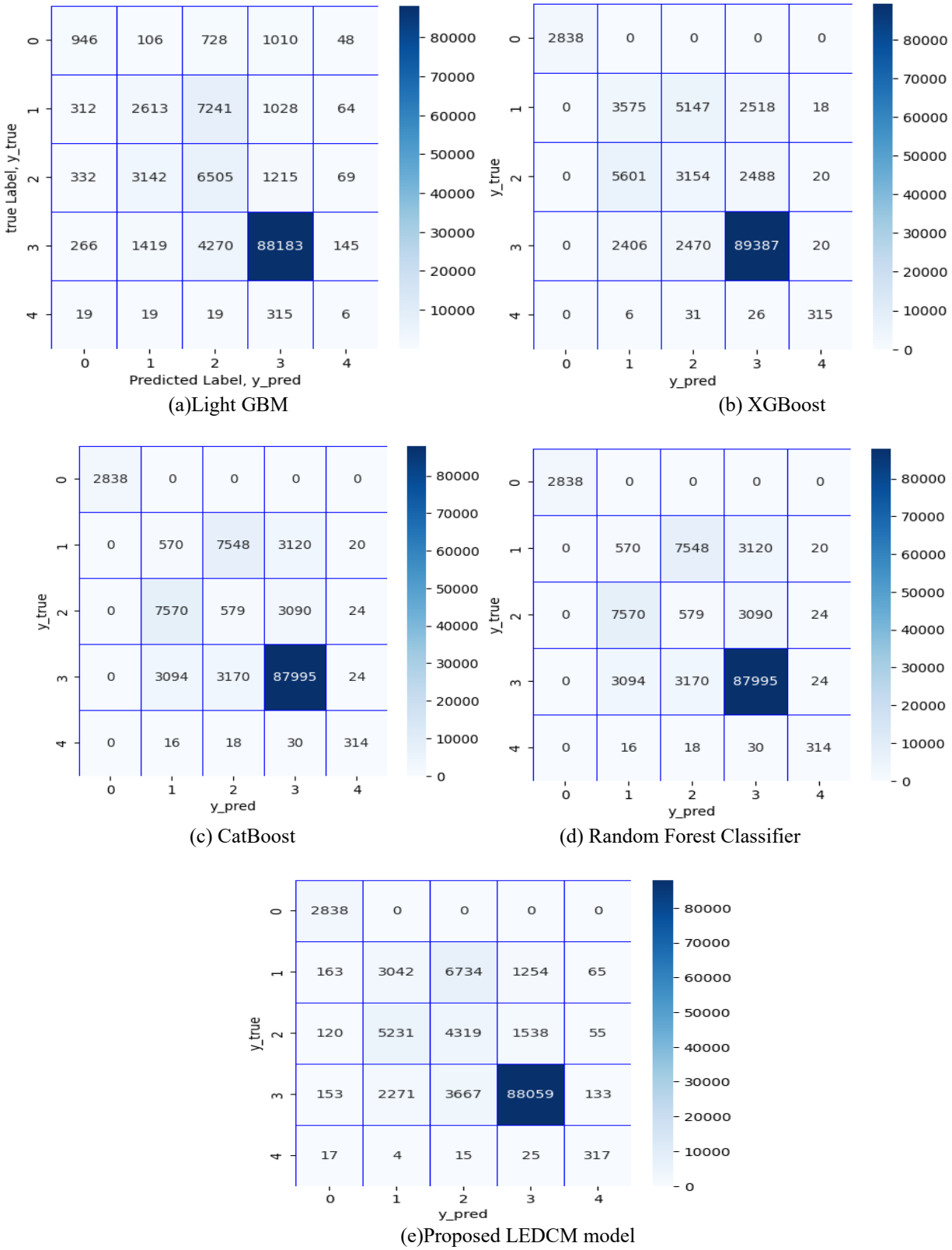
(b) XGBoost

(c) CatBoost

(d) Random Forest Classifier

(e)Proposed LEDCM model

Fig. 13. UNSW-NB15 confusion matrix (Multiclass classification)



(a)Light GBM (b) XGBoost

(c) CatBoost (d) Random Forest Classifier

(e)Proposed LEDCM model

Fig. 14. NF-BOT-IOT confusion matrix (Multiclass classification)

Table 11. Evaluating the performance of other ensemble models with our proposed(NSL-KDD99)

Author	(Sabhnani et al.,2003)	(Hwang et al., 2007).	(Khan et al.,2007)	(NgambaThockchom et al., 2023)	Proposed Model
Classifier(s)	ANN, Gaussian classifier, k-means clustering	MD, AD, SVM	SVM, clustering	DT, LR, GNB, SGD	LightGBM, XGBoost, CatBoost, RFC(Ensemble)
Breach type	Accuracy (%)	Accuracy (%)	Accuracy (%)	Accuracy (%)	Accuracy (%)
Probe	88.70	99.16	97	99.98	1.0
DoS	97.30	97.65	23	99.89	1.0
U2R	29.82	76.32	43	98.81	0.94
R2L	09.60	46.53	91	60	0.99

Table 12. Evaluating the performance of other ensemble models with our proposed using precision and Recall parameters(UNSW-NB15)

Author	(Rajagopal et al.,2020)		(Khammassi et al., 2017)		(NgambaThockchom et al., 2023)		Proposed Model	
Classifier(s)	kNN, SVM, RF, LR		GA, DT		DT, LR, GNB, SGD		LightGBM, XGBoost, CatBoost, RFC (Ensemble)	
Breach type	Prec (%)	Recall	Prec (%)	Recall	Prec (%)	Recall	Prec (%)	Recall
Analysis	67.44	11	44.44	9.92	98.80	98.11	0.73	0.88
Backdoor	70	10.79	51.51	6.92	31.40	08.41	0.44	0.36
DOS	41.60	25	36.0	4.11	61.14	86.20	0.61	0.36
Exploits	63.40	85	60.2	92.31	60.14	61.30	0.89	0.93
Fuzzers	64.40	60.97	70.3	69.11	91.71	91.00	0.95	0.95
Generic	99.40	98.32	99.7	97.93	54.55	19.35	1.00	1.00
Normal	91.67	91.82	92.3	90.72	55.81	13.82	1.00	1.00
Reconnaissance	90.65	74.8	90.88	76.15	47.56	07.98	0.60	0.61
Worms	57.69	37.5	46.8	38.46	86.01	75.55	0.67	0.67

The comparison of the proposed model's performance with other ensemble models on NSL-KDD99 and UNSW-NB15 datasets can be seen in Tables 11 and 12.

Conclusion and Future Work

The major goal of our proposed work or model is to provide a good ensemble framework for IDS prevention in IoT networks. The common IoT dataset, which accurately reflects modern-day traffic, is used in this work (NSL-KDD, UNSW-NB15, NF-BOT-IOT datasets). We proposed an ensemble classifier, where the features are cleaned and transformed, encoded, and then, after all the four base classifiers (LightGBM, XGBoost, Catboost, RFC) further assess the dataset and make predictions with specific accuracy and then after, we will find the Leader Models for each category of Attacks. The proposed method generates a Meta classifier module that will take the best predictions made by base models, specifically for

a category of Attacks, and merge them in common to make the best prediction Model.

After merging the results in common, better results were achieved. The strategies were chosen because they are widely used in the domain, and we used four metrics to assess the results, namely Accuracy, precision, Recall percentage, and F1-score. The IoT or network-based dataset was tested on data samples, with a decompose ratio of 80. 20 80 % of data records are used for training purposes of frameworks (i.e., models or classifiers), and the rest 20 % for testing purposes. The result values show that our proposed model got the highest accuracy of 99.94% for NSL-KDD test data, 95.99% for UNSW-NB-15, and 83.13% for NF-BOT-IOT for Multiclass classification, and an Accuracy of 100% for all three Datasets in case of binary label classification in comparison to other base classifiers. This study will provide valuable insights into efficient intrusion detection techniques specifically designed for IoT networks. By addressing the challenges associated with implementing IDS in IoT, it aims to promote the development and adoption of robust security measures to defend IoT-enabled devices or networks from security threats. Future work could focus on further optimizing the model and exploring additional ensemble techniques to enhance its performance and adaptability in dynamic IoT environments.

Conflicts of Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Availability of Supporting Data

The publicly available datasets used in this study are:

1) For NSL-KDD use URL: https://www.unb.ca/cic/datasets/nsl.html
2) For UNSW-NB15: URL:https://research.unsw.edu.au/projects/unsw-nb15-dataset
3) For NF-BOT-IOT: URL:https://staff.itee.uq.edu.au/marius/NIDS_datasets/#RA2

The datasets generated during and/or analyzed during the current study are not publicly available but are available from the corresponding author on reasonable request.

Authors contributions

Neeraj Sharma: Conceptualization; Methodology; Formal analysis; Investigation; Data analysis and interpretation; Software implementation; Critical revision of the manuscript; Provision of data and resources.

Neelu Nihalani: Conceptualization; Methodology; Formal analysis; Critical revision of the manuscript; Supervision; Validation; Project administration; Provision of data, resources, and analysis tools.

References

Aksu, D., & Ali Aydin, M. (2018, December). *Detecting port scan attempts with comparative analysis of deep learning and support vector machine algorithms*. In 2018 International Congress on Big Data, Deep Learning and Fighting Cyber Terrorism (IBIGDELFT) (ANKARA, Turkey). https://doi.org/10.1109/ibigdelft.2018.8625370

Alazab, M., Abu Khurma, R., Castillo, P. A., Abu-Salih, B., Martín, A., & Camacho, D. (2024). An effective network intrusion detection approach based on hybrid Harris Hawks and multi-layer perceptron. *Egyptian Informatics Journal, 25*(100423), 100423. https://doi.org/10.1016/j.eij.2023.100423

Alanazi, S., Al-Muhtadi, J., Derhab, A., Saleem, K., AlRomi, A. N., Alholaibah, H. S., & Rodrigues, J. J. P. C. (2015, October). *On the resilience of Wireless Mesh routing protocol against DoS attacks in IoT-based ambient assisted living applications*. In 2015 17th International Conference on E-Health Networking, Application & Services (HealthCom) (Boston, MA). https://doi.org/10.1109/healthcom.2015.7454499

Aljuaid, W. H., & Alshamrani, S. S. (2024). A deep learning approach for intrusion detection systems in cloud computing environments. *Applied Sciences, 14*(13), 5381. https://doi.org/10.3390/app14135381

Attou, H., Guezzaz, A., Benkirane, S., Azrour, M., & Farhaoui, Y. (2023). Cloud-based intrusion detection approach using machine learning techniques. *Big Data Mining and Analytics, 6*, 311–320.

Dawoud, A., Shahristani, S., & Raun, C. (2018). Deep learning and software-defined networks: Towards secure IoT architecture. *Internet of Things, 3–4*, 82–89. https://doi.org/10.1016/j.iot.2018.09.003

Diro, A., & Chilamkurti, N. (2018). Leveraging LSTM networks for attack detection in fog-to-things communications. *IEEE Communications Magazine, 56*(9), 124–130. https://doi.org/10.1109/mcom.2018.1701270

Farhan, R. I., Maolood, A. T., & Hassan, N. F. (2020). Optimized deep learning with binary PSO for intrusion

detection on CSE-CIC-IDS2018 dataset. *Journal of Al-Qadisiyah for Computer Science and Mathematics, 12*(3). https://doi.org/10.29304/jqcm.2020.12.3.706

Grace, M., & Sughasiny, M. (2022). Malware detection for Android applications using Aquila optimizer and hybrid LSTM-SVM classifier. *ICST Transactions on Scalable Information Systems*, e1. https://doi.org/10.4108/eetsis.v9i4.2565

Granjal, J., Monteiro, E., & Sa Silva, J. (2015). Security for the Internet of Things: A survey of existing protocols and open research issues. *IEEE Communications Surveys & Tutorials, 17*(3), 1294–1312. https://doi.org/10.1109/comst.2015.2388550

Hwang, T. S., Lee, T.-J., & Lee, Y.-J. (2007, June 12). *A three-tier IDS via data mining approach*. In Proceedings of the 3rd Annual ACM Workshop on Mining Network Data (San Diego, CA). https://doi.org/10.1145/1269880.1269882

Irwin, L. (2024). *Data breaches and cyber attacks quarterly review: Q1 2024*. IT Governance. https://www.itgovernance.co.uk/blog/data-breaches-and-cyber-attacks-quarterly-review-q1-2024

Khan, L., Awad, M., & Thuraisingham, B. (2007). A new intrusion detection system using support vector machines and hierarchical clustering. *The VLDB Journal, 16*(4), 507–521. https://doi.org/10.1007/s00778-006-0002-5

Khammassi, C., & Krichen, S. (2017). A GA-LR wrapper approach for feature selection in network intrusion detection. *Computers & Security, 70*, 0167–4048. https://doi.org/10.1016/j.cose.2017.06.005

Kiflay, A., Tsokanos, A., Fazlali, M., & Kirner, R. (2024). Network intrusion detection leveraging multimodal features. *Array, 22*(100349), 100349. https://doi.org/10.1016/j.array.2024.100349

Lalduhsaka, R., Bora, N., & Khan, A. K. (2022). Anomaly-based intrusion detection using machine learning. *International Journal of Information Security and Privacy, 16*(1), 1–15. https://doi.org/10.4018/ijisp.311466

Montiel, J., Halford, M., Mastelini, S. M., Bolmier, G., Sourty, R., Vaysse, R., Zouitine, A., Gomes, H. M., Read, J., Abdessalem, T., & Bifet, A. (2021). River: Machine learning for streaming data in Python. *Journal of Machine Learning Research*. https://doi.org/10.1109/COMST.2019.2896380

Ngamba Thockchom, N., Singh, M. M., & Nandi, U. (2023). A novel ensemble-based model for network intrusion detection. *Complex & Intelligent Systems*. https://doi.org/10.1007/s40747-023-01013-7

Parampottupadam, S., & Moldovann, A.-N. (2018, June). *Cloud-based real-time network intrusion detection using deep learning*. In 2018 International Conference on Cyber Security and Protection of Digital Services (Cyber Security) (Glasgow). https://doi.org/10.1109/cybersecpods.2018.8560674

Rajagopal, S., Kundapur, P. P., & Hareesha, K. S. (2020). A stacking ensemble for network intrusion detection using heterogeneous datasets. *Security and Communication Networks, 2020*, Article 4586875. https://doi.org/10.1155/2020/4586875

Roopak, M., Tian, G. Y., & Chambers, J. (2020, January). *An intrusion detection system against DDoS attacks in IoT networks*. In 2020 10th Annual Computing and Communication Workshop and Conference (CCWC) (Las Vegas, NV). https://doi.org/10.1109/ccwc47524.2020.9031206

Sabhnani, M., & Serpen, G. (2003). Application of machine learning algorithms to KDD intrusion detection dataset within misuse detection context. *International Conference on Machine Learning: Models, Technologies and Applications*.

Saini, N., Bhat Kasaragod, V., Prakasha, K., & Das, A. K. (2023). A hybrid ensemble machine-learning model for detecting APT attacks based on network behavior anomaly detection. *Concurrency and Computation: Practice and Experience, 35*(28), e7865. https://doi.org/10.1002/cpe.7865

Shenfield, A., Day, D., & Ayesh, A. (2018). Intelligent intrusion detection systems using artificial neural networks. *ICT Express, 4*(2), 95–99. https://doi.org/10.1016/j.icte.2018.04.003

Tsogbaatar, E., Bhuyan, M. H., Taenaka, Y., Fall, D., Gonchigsumlaa, K., Elmroth, E., & Kadobayashi, Y. (2021). DeL-IoT: A deep ensemble learning approach to uncover anomalies in IoT. *Internet of Things, 14*(100391), 100391. https://doi.org/10.1016/j.iot.2021.100391

Wang, S., Xu, W., & Liu, Y. (2023). Res-TranBiLSTM: An intelligent approach for intrusion detection in the Internet of Things. *Computer Networks, 109982*, 109982. https://doi.org/10.1016/j.comnet.2023.109982

Yang, Y., Wu, L., Yin, G., Li, L., & Zhao, H. (2017). A survey on security and privacy issues in Internet-of-Things. *IEEE Internet of Things Journal, 4*(5), 1250–1259. https://doi.org/10.1109/jiot.2017.2694844

Zarpelao, B. B., Miani, R. S., Kawakani, C. T., & de Alvarenga, S. C. (2017). A survey of intrusion detection in the Internet of Things. *Journal of Network and Computer Applications, 84*, 25–37. https://doi.org/10.1016/j.jnca.2017.02.009