Journal of Physics:
Complexity

PURPOSE-LED PUBLISHING™

DDoS Attack Detection in Edge-IIoT Network Using Ensemble Learning

To cite this article before publication: Fariba Laiq *et al* 2024 *J. Phys. Complex.* in press https://doi.org/10.1088/2632-072X/ad506b

IOP Publishing Journal Title
Journal **XX** (XXXX) XXXXXX https://doi.org/XXXX/XXXX

DDoS Attack Detection in Edge-IIoT Network Using Ensemble Learning

Fariba Laiq[1], Feras Al-Obeidat[2], Adnan Amin[3,*], and Fernando Moreira[4]

[1,3]School of Computer Science and Information Technology, Institute of Management Sciences, Peshawar, Pakistan
[2]College of Technological Innovation. Zayed University, Abu Dhabi, UAE.
[4]REMIT, Universidade Portucalense, Porto & IEETA, Universidade de Aveiro, Portugal.
* Correspondence Authors: adnan.amin@imsciences.edu.pk

Received xxxxxx
Accepted for publication xxxxxx
Published xxxxxx

Abstract

As the number of IoT devices increases daily due to the rapid growth in technology, every device and network is vulnerable to attacks because it is exposed to the internet. Denial of Service (DoS) is a prevalent type of intrusion on the Internet of Things (IoT) network in which the server becomes down due to flooding requests. Distributed Denial of Service (DDoS) is a special type of DoS attack where the network of malicious computers called botnet consumes the target's system resources by flooding the requests. Edge computing is closely related to Industrial Internet of Things (IIoT), and industry 4.0. Both of them are relatively emerging technologies so security is a crucial part of them. By incorporating our contributions to the current and innovative dataset Edge-IIoT, the proposed study presents a novel approach to detect DDoS attacks in an IIoT network in the domain of edge computing, whether the traffic is normal or malicious (DDoS traffic). This study explores various Ensemble Learning (EL) techniques to predict normal and malicious DDoS traffic along with the type of DDoS attack. The study applies various preprocessing techniques like Synthetic Minority Over Sampling Technique (SMOTE), label encoding, etc. to enhance the model's performance and reveals how EL techniques performs better in terms of accuracy than the individual classifiers. Further, the performance of all EL techniques has been investigated in terms of all evaluation measures including the elapsed time. This important addition not only broadens the focus of study in this area but also offers insightful comparisons of the efficiency and precision of various ensemble approaches as well as individual classifiers. The study achieved a maximum of 99.99% in all evaluation measures.

Keywords: *DDoS Attack, Edge Computing, Machine Learning, Intrusion Detection, Ensemble Learning, Industrial Internet of Things*

1. Introduction

The IoT is an advanced technology of gathering and transporting data without human involvement. It is a network of objects that contain sensors and software. Recently, the IoT has risen to the forefront of technology, with extensive application in numerous areas and businesses, such as smart homes, smart cities, horticulture, transportation, healthcare, and the armed forces. Because of this, the IoT devices continuously and irreversibly alter our environment and society [1]. In 2025, the International Data Corporation predicted Relative to the anticipated 8.1 billion people, there will be 41.6 billion internet-enabled gadgets generating 79.4 ZB of data [2].

As the number of IoT devices grows, this technology is facing several challenges. Since IoT devices are easily accessible, intruders can utilize them to target the back-end infrastructure to which they are attached [3]. On other hand, with the advancement of technology, the number of attackers has multiplied. Recently, there has been a significant increase in cyber-attacks, which have had a substantial economic effect on businesses relying on computer networks [4]. Numerous types of attacks have been used to create disruptions in networks. DoS attacks are the most prevalent among these. In the past decade, the frequency of such cyber-attacks has increased, establishing them as a major threat to network stability due to their capacity to interrupt various services [5]. DDoS attacks are a common method of deploying this assault. A large network of compromised computers, known as a

botnet, is used to launch a major attack all at once. The goal is to block legitimate access to the system's resources and to reduce the system's performance. [6]. After being discovered in September 2016 by the malware research team, "Malware Must Die," the Mirai virus has drawn notice for its usage in destructive and devastating DDoS attacks [7]. The largest DDoS attack in history was brought on by insecure IoT and DDoS in late 2016. The year of Mirai is (and will continue to be) known as 2016. It exploited them on October 21 to perform the most powerful DDoS attack ever, striking 1.2 terabits per second [8] [9]. DDoS attacks are divided into volumetric attacks and application-layer attacks [10]. The network infrastructure's full bandwidth is used by the flooding assault. It often employs layer three or four protocols to produce huge traffic volumes, such as Internet Control Message Protocol (ICMP), User Datagram Protocol (UDP), and Transmission Control Protocol-Synchronize (TCP-SYN) flood. The Open System Interconnection (OSI) first-layer attack is more complex, and in most circumstances, less bandwidth is used at first. It prioritizes apps or services and progressively depletes network resources. Attackers that supply data with a small packet window can make the connection last longer. There are two types of attacks: Hyper-Text Transfer Protocol (HTTP) and Domain Name Server (DNS) [11].

An Intrusion Detection System (IDS) is a software tool that monitors network activities and detects intrusions that could compromise the system's confidentiality, availability, and integrity. [12]. Conventional signature-based IDS are ineffective for the IoT because of their limited processing, storage, communication methods, and huge data. Developers must improve IoT device detection methods to reduce traffic anomalies. Anomaly-based machine learning (ML) methods, including IoT networks, have successfully detected network abnormalities. ML-based algorithms analyze normal and abnormal IoT traffic [13]. As a result, advanced ML based solutions are better suited to detecting and mitigating the consequences of cyber-attacks [14]. Numerous ML techniques have also been developed as a result of emerging research interests. For instance, Fuzzy Logic (FL), Artificial Neural Networks (ANN), Support Vector Machine (SVM) [15], K-Nearest Neighbor (KNN) [16], Naïve Bayes (NB), and Random Forest (RF) [17][18], to create IDS to detect and prevent the rising cyber-attacks [19].

To ensure the security of IoT/IIoT systems, it's essential to use datasets that accurately represent real-world IoT/IIoT applications. However, many types of research are done on DDoS attack detection in IoT through various techniques and datasets. Techniques Such as FL, ANN, SVM, KNN, NB, and RF considered to design IDS against cyber-attacks on existing datasets. However, there is a recent technology known as Edge-computing. As the number of edge devices increases, it is challenging to tackle the attacks on those devices and networks. Tackling attacks on those devices and networks is challenging.

Edge IIoT is a contemporary, realistic cyber security dataset. A complicated seven-layer testbed that had more than 10 IoT devices, IIoT-based Modbus flows, and 14 attacks connected to the protocol were gathered. This study describes the dataset and its properties [20].

1.2 Motivation

The volume and complexity of IoT networks have increased with the passage of time, especially with the emerging era of Industry 4.0. However, this expansion also gave rise to several security challenges. IoT devices are vulnerable to a variety of security risks, including DDoS attacks, as they are easy to launch as compared to other kind of attacks especially in the context of edge computing in IIoT. These assaults can heavily exploit the security of the entire system with crucial industrial operations as well as endanger data integrity. DDoS assaults increased by 150% during the course of the previous year, with the Americas seeing an astounding 212% spike. Notably, more than half of the assaults targeted organizations in EMEA. Global assault frequency increased by 3.5 times to 29.3 attacks per day, with EMEA seeing an even more concerning rate of 45 attacks per day. The overall number of attacks worldwide increased by 32% from 2021 to 4.44 PB, while the biggest assault ever seen in 2022 was a massive 1.46 Tbps. Globally, 53% of attack activity was directed at the financial sector, which was followed by 20% towards technology and 11% towards healthcare. Experts believe that as 2023 approaches, smaller yet more frequent attacks will increase along with organized and well-funded cybercriminal activities, placing organizations at more danger as they work to protect against the always changing DDoS threat scenario [21].

To retain the secure and dependable functioning of IIoT networks, it is essential to address these security issues. The study addresses this issue of the DDoS attacks in Edge-IIoT by utilizing various ML techniques for the detection and classification of DDoS assaults in edge IIoT networks. The research aims to improve the security measures of edge IIoT networks, using the efficient defense mechanisms against DDoS assaults and advancing safe and resilient industrial systems. Through various approaches of EL techniques like bagging, boosting, hard voting, soft voting, and stacking, the detection of assaults in IIoT will be done by utilizing different preprocessing techniques to enhance the model's performance. Also, the study investigates how EL models perform better than individual classifiers by combining the strengths of each individual classifier and reducing the

likelihood of incorrect results due to the poor performance of an individual classifier.

1.3 Problem Statement

Edge computing and Industry 4.0 are recent advancements in technology. It becomes more difficult to defend against attacks on those devices and networks as the number of edge devices grows. DDoS attacks are a common assault in the IIoT. Only a few studies combined IIoT with edge computing to examine this problem. As we know that EL techniques performs better than the individual classifiers. However, still none of previous studies (discussed in the literature review) in the domain of Edge computing combines with IIoT, has applied a variety of EL approaches along with the time elapsed factor, while taking the data imbalance in account. Therefore, there is a need for a hybrid approach to provide a solution to this issue. As a solution, we will use a variety of EL techniques on a novel, emerging, and real-world dataset, Edge-IIoT, unlike previous research that used old and outdated datasets like KDD-CUPP99, NSL-KDD, BoT-IoT, and TON-IoT etc. The issue of detecting the DDoS attacks through the utilization of multiple EL techniques has been addressed, along with data balancing to avoid the biasness of model. This significant contribution not only expands the scope of research in this domain but also provides valuable insights into the comparative performance of individual classifiers as well as multiple EL techniques in terms of time and accuracy.

1.3 Goals and Objectives

The goal of this research is to design an IDS to detect DDoS attacks in IIoT network in edge computing using various EL approaches in order for the cyber security experts to timely detect the attacks and mitigate them.

The objectives of this research are as follow:

- To shield the Edge-IIoT network against many types of DDoS attacks.
- To reduce the likelihood of false positives in DDoS attacks detection.
- To investigate and compare the performance of multiple EL techniques in terms of accuracy, precision, recall, F1-Score, and elapsed time, to find out which classifier outperforms the other.

2. Literature Review

This section presents an important aspect of what is already known in the target domain implemented by the previous authors along with their techniques, and outcomes.

The IoT is responsible for the development of data streaming from the edge of networks, which has given rise to the edge computing paradigm. The growing sensing capabilities at the edge have created a significant problem for traditional Cloud computing models, particularly with regard to Round Trip Latency time (RTT), which may be high from the edge to the Cloud. For latency sensitive IoT applications like demand forecasting for Smart Power Grids and speech recognition in programs like Apple Siri, this results in a performance trade-off. Edge computing has evolved as a different design paradigm to handle this problem. It uses edge devices for some of the processing and analytics while using the cloud for coordination and data storage. Industry 4.0 and IIoT applications, where real-time response and low latency are essential for effective and intelligent infrastructure management, are particularly crucial for this transition towards processing at the edge. This integration of edge computing with IIoT settings not only improves efficiency but also results in a considerable decrease in power usage when compared to conventional Cloud computing systems [22].

Attacks against edge computing infrastructures have increased significantly in recent years. The Mirai virus, which is a kind of a DDoS attack, was launched in August 2016, when it quickly infiltrated more than 65,000 IoT devices in only 20 hours by taking advantage of their authentication flaws. The infected devices were used to perform large DDoS assaults against edge servers after this initial infection, which knocked down more than 178,000 domains. IoT botnet assaults have resulted in damages surpassing $100 million US since the discovery of the first Mirai botnet in 2016. These occurrences highlight the rising danger that such assaults pose to the reliability and security of edge computing system [23].

In DDoS ICMP attack, the attacker bombards the target system with ICMP echo requests, causing the services used by other applications on the victim's server to break down. The attacker rapidly sends the victim a continuous stream of echo requests to carry out the attack. Another type of assault is a DDoS UDP flood attack, in which the attacker floods the victim's system with a large volume of traffic that contains UDP datagrams. This attack uses fake source IP addresses when coming from a single host. A buffer overflow issue may arise as a result of the excessive number of datagrams that arrive at the victim's server. Another attack known as DDoS TCP SYN flood attack, in which the attacker uses spoof IP addresses to flood the victim's system with bogus SYN requests. The victim never receives more answers to its SYN/ACK packets since the IPs are counterfeit, and as a result, the related ports stay open unnecessarily. All of the victim's ports are blocked when there are a lot of manufactured SYN requests, which prevents real users from connecting The DDoS HTTP attack includes sending a large number of fake HTTP requests from different random IP addresses to the target server. The first HTTP communication connection is never established since the IPs are spoofs, resulting in a

protracted period of false contacts with the server. As a result, the server is unable to accept connections from new, legitimate users [24].

Without Ensemble Techniques

Andrés Chartuni and José Márquez [25] proposed a multi-classifier of DDoS Attacks in networks. On the Canadian Institute for Cybersecurity DDoS 2019 (CICDDoS2019) dataset, they used NN to classify several forms of DDOS assaults and benign traffic. During the data preprocessing stage, they applied multiple techniques to increase the accuracy of results and decrease the training time. Like discarding the input containing missing/infinite values, label encoding using one-hot-encoder, data normalization using the L2 method, and data balancing using SMOTE. Three distinct scenarios were used to train and assess their model. Their model achieved up to 94% of accuracy. Average precision, recall, and F1-score were 94.21%, 94.03%, and 94.12%, respectively.

Mahmoud Said Elsayed et al. [26] proposed DDoSNet (Deep-Learning Model for Detecting Network Attacks). Their method utilized DL combined with the RNN with an auto encoder and softmax regression model at the output layer to classify the network traffic into malicious or normal. They evaluated their model using the CICDDoS2019 dataset. Their model achieved up to 99% of accuracy. The precision on Normal and attack traffic was 100% and 99%, respectively. A recall of 99% was achieved in both classes. F1-score was 99% on both normal and attack traffic.

K.Gurulakshmi et al. [27] proposed a model that aims to distinguish between the normal and abnormal traffic's flow using SVM and KNN to predict earlier abnormal activity. They generated DDOS traffic from several source addresses to a single destination address using the XOIC tool. Using wireshark thy captured the packets. Then using packet analyzer they filtered the packets on the basis of protocols. According to their experiment, their result yields 95% and 90% accuracy for SVM and KNN, respectively, on large feature datasets. In the case of the less featured dataset, SVM and KNN resulted in 97% and 98% accuracy, respectively.

Suleman Mohammed [28] implemented a technique used to distinguish benign traffic from a DDoS assault using DT, KNN, and NB. From the CIC2019DDoS dataset, about nineteen distinct features were carefully picked. The DDoS attack methods employed in the experiment were UDP, DNS, SYN, and NetBIOS. The experiment's findings show that DT and KNNs are the most successful, with 100% and 98% accuracy rates, respectively. With a 29 percent accuracy, NB produced a fairly lousy result. Precision for DT, KNN and Naïve Bayes was 100%, 96%, and 27% respectively. For DT, KNN, and naïve Bayes, the recall was 100%, 97%, and 100%, respectively. F1-score for DT, KNN and NB were 100%, 97%, and 42% respectively.

Ioana Apostol et al. [29] proposed an IDS for IoT botnet anomaly detection. They used unsupervised DL, based on a deep auto encoder model. Dimensionality reduction, data balancing, feature extraction, and deep autoencoder applied on Bot-IOT dataset. Their study evaluated their capacity to detect threats using balanced and imbalanced datasets. According to the findings of their experiments, they achieved 99.7% accuracy, 99% for precision, and 99% for recall.

Xavier Larriva-Novo et al. [30] Implemented an IDS for IoT using preprocessing characterization on UGR16, UNSW-NB15, and KDD99 datasets using a multi-layer perceptron NN. They studied and evaluated several preprocessing techniques for a ML NN algorithm based on traffic categorization. Their study's most relevant conclusion is the importance of preprocessing characteristics, such as basic characteristics and statistical traffic characteristics, using z-score standardization techniques, which allows for increased precision since it allows using the mean deviation of the variables. By applying the categorization of network traffic and several preprocessing methods, accuracy enhanced up to 45%. The results of their model are summarized in table 1.

Jingyi Su et al. [31] presented investigated how ML algorithms like DT, RF, and GBM predict IoT devices attacks using IoT 2020 dataset. They concluded that a DT algorithm is generally more accurate than the RF and GBM. However, the RF achieved better AUC scores because it combines the results of multiple individual DT's. The GBM works well, but it may not be the ideal solution in terms of accuracy and time. The results of their model are summarized in table 1.

Imtiaz Ullah et al. [32] proposed a hybrid DL model for anomaly detection in IoT networks. Their model classifies binary and multi-class IoT network data using CNN and GRU on BoT-IoT, IoT network intrusion, MQTT-IoT-IDS2020, IoT-23, IoT-DS-1, and IoT-DS-2 dataset. Results showed that CNN-GRU achieved 99.92% accuracy for multi-class classification. For binary classification, they achieved an accuracy of 99.96%, precision of 99.90%, recall of 99.95%, and F1-score of 99.93%. In the case of multi-class classification, their model achieved an accuracy of 99.92%, precision of 99.91%, recall of 99.91%, and F1-score of 99.91%. However, in preprocessing techniques the issue of class imbalance and its solution was not discussed.

Tarun Ganesh Palla and Shahab Tayeb [33] presented a technique where an exact number of hidden layers and neurons were used in their study to prevent the problem of over fitting. The dataset was created by integrating Mirai and benign datasets of the Mirai malware detection, referring to seven IoT devices. This research presents a comparative

analysis of the dataset's ANN and Random Forest models. The dataset utilized in this study is the "N-BaIoT" dataset. The best performance was reached with 92.8 percent accuracy, a FNR of 0.3%, and F1-score of 99%.

Mona Esmaeili et al. [34] proposed a study, focused on meta-innovations in DL-based IDS. The research investigates the performance of various ML algorithms, including MLP, LSTM, BiLSTM, KNN, SVM, LDA, DT, and RF, on NSL-KDD dataset, to detect DDoS attacks in IoT. The accuracy values of MLP, LSTM, BiLSTM, KNN, SVM, LDA, DT, and RF are reported as 79.5%, 80%, 82.3%, 77%, 82.8%, 69%, 77.7%, and 75.4%, respectively. The results indicated that BiLSTM approach accurately detects and classifies the DDoS attacks, with an accuracy of 82.3%. Sequential models like LSTM and BiLSTM demonstrated effectiveness in identifying brutal attacks in multiclass classifiers, while BiLSTM stands out as the most reliable solution for evaluating DDoS attacks in IoT.

Md Mamunur Rashid et al. [35] in their research presented a FL approach for detecting intrusions in IoT networks. Their proposed approach involves local IoT clients training their respective data and exchanging parameter updates with a central global server. This facilitates the distribution of an enhanced detection algorithm while preserving the privacy of IoT devices. The authors evaluate their approach using the Edge-IIoTset dataset and utilized DL classifiers, specifically CNN and RNN. Experimental results demonstrate an accuracy of 92.49%, indicating the effectiveness of the FL-based intrusion detection model.

Mohammed Hasan Ali et al. [36] proposed an IDS by utilizing an IGA-BP network, a combination of an Improved GA and Backpropagation NN incorporating an autoencoder network model and an improved genetic algorithm for intrusion detection. The system achieved a detection rate of 98.98% and an accuracy of 99.29% with minimal processing complexity. By employing evolutionary sparse convolution networks and training patterns, the proposed IDS classified normal and malicious activities in the IoT. The authors utilized the dataset CIC-DDoS2019. The limitations included ensuring high reliability, fast computation, and reduced complexity.

Kuburat Oyeranti Adefemi Alimi et al. [37] addressed the need for an improved IDS capable of handling voluminous datasets in IoT networks. They proposed a refined LSTM model tested on two datasets, NSL-KDD and CICIDS-2017, after applying preprocessing techniques such as encoding, dimensionality reduction, and normalization. On CICIDS-2017 dataset, their model achieved 99.22% accuracy, 99.23% precision, 99.22% recall, and 99.22% F1-score for identifying DoS assaults. Similarly, the RLSTM model, when put to the test on the NSL-KDD dataset, obtained 98.60% across all evaluation metrics. Experimental results demonstrated that the RLSTM model outperformed other ML methods, achieving high accuracy rates and superior intrusion detection performance.

Josue Genaro Almaraz-Rivera et al. [38] Presented a study focused on evaluating the impact of record timestamps on prediction accuracy and utilized three different feature sets for binary and multiclass classifications. Their research addressed the dataset bias problem in the Bot-IoT dataset without synthetic data generation or class weights. By avoiding feature dependencies their research achieved an accuracy of over 99%. The DT and Multi-layer Perceptron models exhibited the best performance in detecting DDoS and DoS attacks. Across three different feature sets, the average accuracy exceeded 99% and achieved 100% accuracy in various combinations of normal flows versus DDoS/DoS subcategories. In the case of multiclass classification, their model achieved 99.945%, 99.89%, and 99.917% using the three feature sets. For binary classification, their model achieved accuracies exceeding 99.85%.

Amina Khacha et al. [39] utilized a hybrid DL model combining CNN and LSTM. The dataset used is Edge-IIoTset, which contains real traffic data from IoT and IIoT applications. The performance of the proposed CNN-LSTM model was assessed in terms of accuracy, precision, FPR, and detection cost for both binary and multi-class classifications. All of their models achieved 100% accuracy and precision in binary classification with a FPR of 0%. Both the CNN-LSTM model and the ML-based model exhibited a false positive rate (FPR) of 0%, while the LSTM model had a slightly higher FPR of 0.016%. In terms of multiclass classification, CNN-LSTM model demonstrated high accuracy and precision across all attack classes. The normal class and MITM class achieved a 100% accuracy rate and a 0% FPR for all models. However, the ML-based models showed poor FPR and precision rates for the injection class. The results demonstrated that the CNN-LSTM model outperformed LSTM and traditional ML models.

Vanlalruata Hnamte et al. [40] presented a NIDS based on (DCNNBiLSTM). The model incorporates a combination of CNN and BiLSTM networks, enhanced with an attention mechanism. The DL model learns the characterization features of high-dimensional data. The performance evaluation is conducted using multiclass classification, and accuracy rates of 100% and 99.64% are achieved when trained and tested on CICIDS2018 and Edge_IIoT datasets.

Farhan Ullah et al. [41] implemented an IDS using transformer-based transfer learning for Imbalanced Network Traffic to address the challenges that arises due to complex features and data imbalance. To solve the problem of imbalanced data they used SMOTE technique on UNSW-NB15, CIC-IDS2017, and NSL-KDD datasets. Their research

utilized a CNN model to extract deep features from the balanced network traffic. Finally, developing a hybrid CNN-LSTM model to detect different types of attacks using these deep features. Their model outperforms baseline methods with precision, recall, f1-score, and accuracy metrics of 99%, 100%, 99%, and 99.21% respectively. Furthermore, an explainable AI experiment is conducted to interpret the designed approach and analyze the most reliable and efficient features contributing to specific attack types.

Firooz B. Saghezchi et al. [42] proposed a ML approach for network anomaly detection, specifically for DDoS attacks in Industry 4.0 Cyber-Physical Production Systems. Unlike previous techniques that used synthetic or small-scale lab testbed data, this study utilizes network traffic data captured from a real-world semiconductor production factory. By extracting 45 bidirectional network flow features, several labeled datasets are constructed for training and testing ML models. The performance of 11 supervised, unsupervised, and semi-supervised algorithms is evaluated through simulations, with the supervised algorithms outperforming the others in terms of detection accuracy. The DT model achieves a high accuracy of 99.9% with a low FPR of 0.1%.

Hakan Can Altunay et al. [43] in their study, focused on developing an IDS for IIoT networks using CNN, LSTM, and a hybrid combination of CNN + LSTM. The UNSW-NB15 and X-IIoTID datasets were used to train and evaluate the models through binary and multi-class classification. The hybrid CNN + LSTM model achieved the highest accuracy in intrusion detection for both datasets. It obtained an accuracy of 93.21% (binary) and 92.9% (multi-class) for UNSW-NB15, and 99.84% (binary) and 99.80% (multi-class) for X-IIoTID. However, further improvements can be made like solving imbalanced data issues and the feature selection to enhance detection accuracy in updated datasets.

Mohamed Amine Ferrag et al. [44] introduced three DL-based IDS models, CNN, DNN, and RNN for cyber security in agriculture 4.0 using CIC-DDoS2019, and TON_IoT datasets. Performance evaluation was performed for both binary and multiclass classifications. The experimental results demonstrated that DL techniques outperformed other ML strategies, and the CNN-based IDS model exhibited superior performance compared to DL IDS methods by achieving an accuracy of 99.95% for binary traffic detection and 99.92% for multiclass traffic detection.

Shahid Latif et al. [45] presented an IDS using a Lightweight Random Neural Network. The study highlights the need for new datasets that cater to the modern security requirements of IIoT networks and discusses the potential deployment of the proposed attack detection system on resource-constrained devices like Raspberry Pi 4 and Intel Neural Computing Stick. Although the proposed model is evaluated using the DS2OS dataset. Their model achieved an accuracy of 99.20% with a learning rate of 0.01 and a prediction time of 34.51 milliseconds. Other evaluation measures like precision, recall and F1-Score achieved 99.11%, 99.13%, and 99.20% respectively.

With Ensemble Techniques

Wenbin Yao et al. [46] presented a light-weight IDS for IoT, incorporating one-class encoder and EL. The authors aimed to address the challenges posed by unknown attacks and the imbalance of attack types in traditional datasets. Their proposed approach utilizes a One-Class Bidirectional GRU Autoencoder for accurate identification of normal, abnormal, and unknown attack data. Additionally, an EL method with soft voting is employed to classify the anomaly types identified by multiple base classifiers, improving the accuracy of recognizing unknown attacks as the most similar known attack type. Experimental evaluations was performed on WSN-DS, UNSW-NB15, and KDD CUP99 datasets demonstrate recognition rates of 97.91%, 98.92%, and 98.23%, respectively.

Vibekananda Dutta et al. [47] proposed using an ensemble method for anomaly-based NIDS, including DL algorithms and the notion of stacking generalization. This research uses several feature engineering approaches combined with dimensionality reduction to attain the best efficiency. Using a combination of Deep Neural Networks (DNN) and Long Short-Term Memory (LSTM) networks, complemented by a meta-classifier, led to significant improvements in performance and anomaly detection. This study utilized the IoT-23, LITNET-2020, and NetML-2020 datasets. They recorded a detection accuracy of 99.7% with the IoT-23 dataset, 100% with the LITNET dataset, and 100% with the NetML-2020 dataset. Specifically, for the IoT-23 dataset, they achieved a precision of 100%, a recall of 95%, and an F1-score of 98%. For the LITNET dataset, the precision, recall, and F1-score all reached 100%. In the case of the NetML-2020 dataset, the precision was 99%, the recall was 99.9%, and the F1-score was 100%.

Although their model achieved a maximum accuracy of 100% but there are two major limitations. The elapsed time for the techniques were not discussed. Secondly, only one ensemble technique, i-e stacking was used. There is still a gap to evaluate the model's performance by applying and comparing multiple ensemble techniques.

Yan Song et al. [48] implemented an IDS using WOA-XGBoost on KDD CUP 99 dataset. They used Whale Optimization Algorithm to discover the optimal parameters for the model. The acquired network data is the first dimensionality reduced using PCA. Then it is loaded into the WOA-XGBoost algorithm to improve the overall model's data

intrusion detection accuracy after training. Their model attained 99% accuracy, 99.5% sensitivity, and 95.4% specificity.

Preethi Devan et al. [49] implemented an efficient IDS that uses the XGBoost algorithm for feature selection and DNN for network intrusion classification. Normalization, feature selection, and classification are the three processes of the XGBoost–DNN model. During DNN training, they used the Adam optimizer to optimize the learning rate and the softmax classifier to classify network intrusions. The research was implemented using Tensor flow and Python on the NSL-KDD dataset. Cross-validation was used to validate the model. With a population size of 7000, the model has obtained maximum accuracy, precision, recall, and F1-score of 97%.

Sumaiya Thaseen Ikram et al. [50] in their study implemented an ensemble of multiple DNN models such as MLP, BPN, and LSTM was stacked in their study. The ensemble model's performance was evaluated using two datasets: UNSW-NB15 and VIT SPARC20, which were created on campus. The VIT SPARC20 dataset also captures other forms of traffic, such as normal unencrypted traffic, normal encrypted traffic, and encrypted and unencrypted malicious traffic. Without decrypting the contents, the DL models categorize encrypted normal and malicious traffic of VIT SPARC20. To obtain improved accuracy, XGBoost combines the findings of each DL model. On UNSW_ NB dataset, the model has achieved maximum accuracy of 99.5%, precision of 99.45%, recall of 99.42%, and F1-Score of 99.5%. The performance on the VIT_SPARC20 dataset in terms of accuracy, precision, and recall was 99.4%, 98%, and 97%, respectively.

Sweta Bhattacharya et al. [51] implemented a model first performs One-Hot encoding for the transformation of the “Intrusion Detection” dataset. For dimensionality reduction, they used the PCA-firefly algorithm. The XGBoost algorithm was implemented on the reduced dataset for classification. The comparative analysis of all the combined approaches was done. Results highlighted that the performance of XGBoost-PCA-firefly outperforms the other ML algorithms. Their model obtained an accuracy of 99.9%, a sensitivity of 99.3%, and a specificity of 99.9%.

Hui Jiang et al. [52] proposed a PSO-XGboost model to optimize the XGBoost parameters for intrusion detection using the NSL-KDD dataset. The XGboost model can be used to solve multi-classification issues. PSO can quickly arrive at an estimated optimum solution. Compared to other models such as RF, Bagging, and Adaboost, the experimental findings reveal that their model has a higher mean average precision and macro value. Their model achieved a true positive rate of 92%. In terms of mean average precision values, the curves of the four models overlap, with PSO-Xgboost having the highest area, 0.64.

Thi-Thu-Huong Le et al. [53] proposed a solution for imbalanced multiclass output data in IIoT datasets and its impact on ML-based IDS for attack detection. The authors propose a novel solution by leveraging the XGBoost model, specifically designed for imbalanced IIoT datasets. Three evaluation measures, precision, recall, and F1-Score were used on two datasets, X-IIoTDS and TON_IoT. On X-IIoTID dataset, they achieved an average F1 score of 99.9% for the two output classes. Similarly, on the TON_IoT dataset, they achieved an average F1 score of 99.87%.

Safdar Hussain Javed et al. [54] proposed an intelligent Advanced Persistent Threat (APT) detection and classification system designed to improve the security of IIoT. The paper applied various ML algorithms, including DT, RF, SVM, LR, GNB, Bagging, XGBoost, and Adaboost. The dataset used is KDDCup99 to detect and classify complex APT signatures after pre-processing the data. To find the best solution for the specified domain, the research also carried out a comparison analysis of the ML algorithms. Experimental results reveal that the Adaboost classifier achieved the best results with an accuracy of 99.9% and an execution time of 0.012 seconds for APT attack detection.

Joseph Bamidele Awotunde et al. [55] proposed an ensemble tree-based model for IDS in IoT. Their study proposed ensemble models empowered with a feature selection classifier for IDS in IIoT networks. Using the Chi-Square Statistical method for feature selection, different ensemble classifiers like XGBoost, Bagging, Extra Trees (ET), RF, and AdaBoost are applied to telemetry data from the TON-IoT datasets. Among the ensemble models evaluated, Bagging achieved an accuracy, precision, recall, and F1-score of 99%, XGBoost demonstrated scores of 100% in all metrics. RF achieved a score of 99% in accuracy, precision, recall, and F1-score. The ET model exhibited similar results, achieving 99% in all evaluation measures except for an F1-score of 98%. However, AdaBoost yielded relatively lower scores, with an accuracy, precision, recall, and F1-score of 70%, 69%, 66%, and 60%, respectively, suggesting a reduced ability to accurately detect and classify IIoT attacks compared to the other models. Their results indicated that XGBoost ensemble outperforms other models in detecting and classifying IIoT attacks. However, the suggested model's failure to address the class disparity is one of its main drawbacks.

Mona Alduailij et al. [56] presented an approach for detecting DDoS attacks in cloud computing with the objective of reducing misclassification errors on CICIDS 2017 and CICDDoS 2019 datasets. Two feature selection techniques, Mutual Information and Random Forest Feature Importance, are employed to select the most relevant features. The selected features are then utilized by ML algorithms RF, GB, Weighted Voting Ensemble, KNN, and LR for DDoS attack detection. Experimental results demonstrate that RF, GB, WVE, and

KNN achieve an accuracy of 99% when using 19 features. The study further analyzes misclassifications to improve measurement accuracy. The performance of RF stands out.

Abdul Rahman al-Abassi et al. [57] implemented a method that involves data balancing using a deep representation-learning model. These representations are then utilized in an ensemble deep learning algorithm consisting of DNN and DT classifiers for attack detection. The performance of the proposed model is evaluated on real ICS datasets. Their approach achieved 95.86% accuracy for the gas pipeline dataset and 99.67% for the secure water treatment dataset.

V. Priya et al. [58] proposed a technique with the objective of developing a two-phase anomaly detection model for enhancing the security of IIoT networks. The first phase integrates SVM, KNN, and NB classifiers using an ensemble blending technique. K-fold cross-validation is performed to optimize the training and test sets. Also, an ANN classifier with the Adam optimizer is used for prediction. In the second phase, the results from both the ANN and RF classifiers are combined in the classification unit. The authors applied the techniques on WUSTL_IIOT-2018, N_BaIoT, and Bot_IoT datasets. Their proposed achieved a maximum accuracy of 99%.

Bidyapati Thiyam, and Shouvik Dey [59] presented a study, focused on solving the challenges due to class imbalance in the CIC-DDoS2019 and Edge-IIoT datasets. To tackle this issue, a hybrid approach combining SMOTE and TOMEK links is proposed. Moreover, a feature shuffle method with RF is used to calculate the ROC values of each feature, eliminating unwanted noise. The top 15 features are selected based on their relevance, forming a subset of features. The model was trained using six ML algorithms, KNN, LR, DT, RF, GDB, and ADB, and the results indicated that DT achieved the highest accuracy of 99.87% for the CIC-DDoS2019 dataset and 99.32% for the ML-EdgeIIoT dataset. All algorithms exhibited accuracy above 90%, except for LR, which had an accuracy of 84.43% for the ML-EdgeIIoT dataset.

The table 1 summarizes the literature review in terms of the authors, dataset, and techniques used and the best outcomes.

Table 1: Related Work

Author [Reference]	**Year**	**Dataset used**	**Technique**	**Best Outcomes**
Andrés Chartuni et al. [25]	2021	CICDDoS2019	NN	Max Accuracy: 94%, Avg precision: 94.21%, Avg recall: 94.03%, F1-score 94.12%.
Mahmoud Said Elsayed et al. [26]	2020	CICDDoS2019	DL	Accuracy: 99%, Precision 100%, Recall: 99%, F1-Score: 99%.

K.Gurulakshmi et al. [27]	2018	Primary Dataset	SVM and KNN	Max Accuracy: 97% for SVM 98% for KNN.
Suleman Mohammed [28]	2021	CICDDoS2019	DT, KNN, and NB	Max Accuracy: 100%, Max Precision: 100%, Max Recall: 100%, Max F1-Score: 100% (with DT)
Ioana Apostol et al. [29]	2021	Bot-IOT dataset	Unsupervised Deep Learning based on a deep autoencoder model	Accuracy: 99.7%, Precision: 99%, Recall: 99%.
Xavier Larriva-Novo et al. [30]	2021	UGR16, UNSW-NB15, and KDD99	Preprocessing Techniques: Basic characteristics and statistical traffic characteristics, using z-score standardization techniques. Algorithm: Multi-layer perceptron neural network	**NSL-KDD dataset:** Max Accuracy: 99.7%, Max Precision: 99.2%, Max Recall: 99.99% **UGR16 dataset:** Max Accuracy: 99.3%, Max Precision: 98.8%, Max Recall: 95.9% **UNSW-NB15 dataset:** Max Accuracy: 99.2%, Max Precision: 97.3%, Max Recall: 99%
Jingyi Su et al. [31]	2022	IoT 2020	DT,RF, and GBM	**DT:** Normal-Abnormal: 96.8%, Mirai-Other: 90.9%, DoS-Other: 99.9%, MITM-Other: 91.5%, Scan-Other: 92.6% **Random Forest:** Normal-Abnormal: 97.8%, Mirai-Other: 90.7%, DoS-Other: 99.9%, MITM-Other: 91.3%, Scan-Other: 92.5% **GBM:** Normal-Abnormal: 96.3%, Mirai-Other: 88.9%, DoS-Other: 99.9%, MITM-Other: 87.3%, Scan-Other: 90%

Imtiaz Ullah et al. [32]	2021	BoT-IoT, IoT network intrusion, MQTT-IoT-IDS2020, IoT-23, IoT-DS-1, and IoT-DS-2	CNN and GNN.	**Binary classification:** Accuracy: 99.96%, Precision: 99.90%, Recall: 99.95%, F1-score: 99.93%. **Multi-class classification:** Accuracy: 99.92%, Precision: 99.91%, Recall: 99.91%, and F1-score of 99.91%.
Tarun Ganesh Palla et al. [33]	2021	N-BaIoT	ANN and RF	Accuracy: 92%, False-negative rate: 0.3%, F1-score: 99%
Mona Esmaeili et al. [34]	2022	NSL-KDD	MLP, LSTM, BiLSTM, KNN, SVM, LDA, DT, and RF	Accuracy: MLP: 79.5%, LSTM: 80%, BiLSTM: 82.3%, KNN: 77%, SVM: 82.8%, LDA: 69%, DT: 77.7%, RF: 75.4%
Md Mamunur Rashid et al. [35]	2023	Edge-IIoTset	CNN, RNN	Best accuracy: 92.49%
Mohammed Hasan Ali et al. [36]	2022	CIC-DDoS2019	IGA-BP	Accuracy: 99.29%
Kuburat Oyeranti Adefemi Alimi et al. [37]	2022	NSL-KDD CICIDS-2017	Refined LSTM	**On CICIDS-2017 dataset:** Accuracy: 99.22% Precision: 99.23%, Recall: 99.22% F1-Score: 99.22% **On NSL-KDD dataset:** Accuracy: 98.60% Precision: 98.60% Recall: 98.60% F1-Score: 98.60%
Josue Genaro Almaraz-Rivera et al. [38]	2022	Bot-IoT	DT, Multi-layer perceptron	Max Accuracy 100%
Amina Khacha et al. [39]	2023	Edge-IIoTset	CNN-LSTM	Accuracy: 100%

Vanlalruata Hnamte et al. [40]	2023	CICIDS2018, Edge-IIoT	DCNNBiLSTM	**On CICIDS2018 dataset:** Accuracy: 100% **On Edge-IIoT dataset:** Accuracy: 99.64%
Farhan Ullah et al. [41]	2023	UNSW-NB15, CIC-IDS2017, and NSL-KDD	CNN-LSTM	Accuracy: 99.21%, Precision:99%, Recall: 100%, F1-Score: 99%
Firooz B. Saghezchi et al. [42]	2022	Primary dataset from semiconductor production factory.	OneR, LR, NB, BN, KNN, SVM, DT, RF, KMeans, EM,	Accuracy: 99.9%, Precision: 99.9%, Recall: 99.9%, FPR: 0.1%
Hakan Can Altunay et al. [43]	2023	UNSW-NB15 and X-IIoTID	CNN+LSTM-	**On UNSW-NB15 dataset:** Accuracy: 93.21% for binary classification and 92.9% for multi-class. **On X-IIoTID:** Accuracy: 99.84% for binary and 99.80% for multi-class.
Mohamed Amine Ferrag et al. [44]	2021	CIC-DDoS2019, and TON_IoT	CNN, DNN, and RNN	**For binary classification:** Accuracy: 99.95% **For multi-class classification:** Accuracy: 99.92%
Shahid Latif et al. [45]	2020	DS2OS	RaNN	Accuracy: 99.20%, Precision: 99.11%, Recall: 99.13%, F1-Score: 99.20%
Wenbin Yao et al. [46]		WSN-DS, UNSW-NB15, KDD CUP99	One-Class Bidirectional GRU Autoencoder Model using EL with soft voting	**On WSN-DS dataset:** Accuracy: 97.91% **On UNSW-NB15:** Accuracy: 98.92% **On KDD CUP99:** Accuracy: 98.23%
Vibekananda Dutta et al. [47]	2020	IoT-23, LITNET-2020, and NetML-2020	DNN and LSTM ensemble through meta-classifier	**IOT-23 dataset:** Accuracy of 99.7%, Precision: 100%, Recall: 98%, F1-Score: 95%. **LITNET dataset:** Accuracy: 100%, Precision: 100%, Recall 100%, F1-Score: 100%. **NetML-2020 dataset:** Accuracy: 100%, Precision: 99%, Recall: 99.9%, F1-Score: 100%.

Yan Song et al. [48]	2022	KDD CUP 99	WOA-XGBoost	Accuracy: 99%, Sensitivity: 99.5%, Specificity: 95.4%
Preethi Devan et al. [49]	2020	NSL-KDD	XGBoost-DNN	Accuracy: 97%, Precision: 97%, Recall: 97%, F1-Score: 97%
Sumaiya Thaseen Ikram et al. [50]	2021	UNSW-NB15 and VIT SPARC20	Ensemble of Multilayer Perceptron, Back Propagation Network, and LSTM through Stacking.	**VIT_SPARC20 dataset:** Accuracy: 99.4%, Precision: 98%, Recall: 97%. **UNSW-NB dataset:** Accuracy: 99.5%, Precision: 99.45%, Recall: 99.42%, F1-Score: 99.52%
Sweta Bhattacharya et al. [51]	2020	Intrusion Detection	PCA-Firefly based XGBoost	Accuracy: 99.9%, Sensitivity: 99.3%, Specificity: 99.9%.
HUI JIANG et al. [52]	2020	NSL-KDD	PSO-XGBoost	True positive rate: 92%. Mean Average precision value: 64%.
Thi-Thu-Huong Le et al. [53]	2022	X-IIoTDS TON_IoT	XGBoost	On X-IIoTDS: F1-Score: 99.9% On TON_IoT: F1-Score: 99.87%
Safdar Hussain Javed et al. [54]	2022	KDD CUP99	DT, RF, SVM, LR, GNB, Bagging, XGBoost, and Adaboost	DT: 96.6%, RF: 98.5, SVM: 98.7%, LR: 98.5%, GNB: 97.2%, Bagging: 97.5%, XGBoost: 98.7%, Adaboost: 99.9%.
Joseph Bamidele Awotunde et al. [55]		TON-IoT	XGBoost, Bagging, Extra Trees, RF, and AdaBoost	**Bagging:** Accuracy: 99% , Precision: 99%, Recall: 99%, F1-score: 99% **XGBoost:** Accuracy: 100%, Precision: 100%, Recall: 100%, F1-score: 100% **RF:** Accuracy: 99% , Precision: 99%, Recall: 99%, F1-score: 99% **F1-score:** Accuracy: 99% , Precision: 99%, Recall: 99%, F1-Score: 98% **AdaBoost:** Accuracy: 70%, Precision: 69%, Recall: 66%, F1-Score: 60%
Mona Alduailij et al. [56]	2022	CICIDS 2017, CICDDoS 2019	RF, GB, Weighted Voting Ensemble, KNN, and LR	Max Accuracy: 99%

Abdul Rahman al-Abassi et al. [57]	2020	Gas pipeline dataset, Secure water treatment dataset	DNN and DT	On gas pipeline dataset: Accuracy: 95.86% On secure water treatment dataset: Accuracy: 99.67%
V. Priya et al. [58]	2020	WUSTL_IIOT-2018, N_BaIoT, Bot_IoT	SVM, k-NN, and NB through stacking EL, DT, RF, ANN classifier with the Adam optimizer	Max Accuracy: 99%
Bidyapati Thiyam, and Shouvik Dey [59]	2023	CIC-DDoS2019 and Edge-IIoT	k-NN, LR, DT, RF, GDB, and ADB.	On CIC-DDoS2019 dataset: Accuracy: 99.87% On ML-EdgeIIoT dataset: Accuracy: 99.32%
Jai Prakash Mishra et. Al. [60]	2023	MEMS sensor and IoT	k-NN, SVM, and Bagged tree	Validation Accuracy: k-NN: 0.9906 SVM: 0.9368 Bagged tree: 0.9887

Limitations of the existing work

From the above literature reviews, some of the limitations identified are, the use of old and outdated datasets which does not contain the recent kind of attacks, not all the EL techniques were applied on any study, elapsed time were not considered of each applied algorithm, and none of the research explored the problem of intrusion detection due to DDoS attacks in the domain of industry 4.0 with Edge computing. The proposed research addresses these issues and fill this gap by securing the Edge network in the domain of IIoT through all the EL techniques with proper preprocessing methods, also taking the elapsed time of each algorithm in account.

3. Preliminary Study

This chapter focuses on the background and explanation of each topic related to this research as it is an important step in comprehending and examining the research problem. This chapter explains the individual classifiers, the ensemble classifiers, and the evaluation measures used.

3.1. Base Classifiers

In the proposed research, base classifiers are the standalone individual classifiers. Later, they are used together using EL techniques. They are used to classify the traffic data as normal or a certain type of DDoS attack.

3.1.1. SVM

The idea behind the support-vector network is to use a specified non-linear mapping to turn input vectors into a higher-dimensional feature space, Z. An established linear decision boundary in this area has special characteristics that improve the network's ability to generalize information effectively. It is a type of classification algorithm in ML having two main benefits over more recent methods, such as neural networks like quicker processing speed and higher performance when working with a small number of samples, often in thousands. In SVM, each data point is modelled as a point in a multidimensional space, where the dimensions correspond to the data's attributes. The method seeks an ideal hyperplane that effectively divides the two groups by visualizing the data in this space. With this hyperplane serving

as the decision boundary, the SVM classifies new instances according to where they are in relation to the hyperplane [61].

3.1.2. DT

The DT is a supervised ML approach used to categories the data. Breiman and his colleagues worked together to create CART, which stands for Classification and Regression Trees. Each internal node of this algorithm's binary tree structure has two outward edges. In DT, the class label prediction process begins at the tree's root. The root attribute's values are compared with that of the record being evaluates attribute. The relevant branch is then followed, leading to the following node in the tree. Based on the tree's path taken, this iterative process continues until a final prediction is made The algorithm finds appropriate splits using the Towing Condition, leading to the creation of the tree [62].

3.1.3. NB

It is a supervised ML algorithm for classification purpose that is based on the Bayes Theorem and assumes independence among each attribute and the class label. When dealing with big datasets and the necessity for speedy model creation, it is very helpful. The algorithm chooses the class with the highest probability as the expected result by computing the conditional probability of a class given the input features. Using the training data, NB calculates the probabilities of each class as well as the probability of various feature values within each class. The possibility of witnessing particular features for each class is represented by a model that is constructed using these probabilities. The approach determines the posterior probability of each class given the observable characteristics when fresh, unseen examples need to be categorized. The class with the highest posterior probability is then given the instance [63].

3.2. EL Techniques

EL is a great approach in ML that combines the results obtained from multiple base classifiers and then makes a prediction. In this way this technique seeks better overall predictive performance by decreasing the likelihood of an unfortunate selection from a poor classifier.

As the ensemble classifier combines the results from multiple base classifiers, there are further various techniques for combining the results [64]. Below are the well-known techniques in EL that are used in my research.

3.2.1. Bagging

Bagging is an EL technique that is used to reduce variance in datasets with noise. It works by picking data points with replacement, allowing for probable recurrence, and generating several random samples from a training set. Then, each of these samples is utilized to separately train a different model. The predictions produced by these models are aggregated by taking the average or majority vote, depending on the relevant job, such as regression or classification, producing a more accurate assessment. By utilizing the variety of several homogenous models trained on various subsets of the dataset, bagging helps to reduce the effects of noisy data. This combination is achieved by averaging the predictions in the case of numerical outcomes or using majority voting for class predictions. Bagging improves the overall performance of the ensemble model by aggregating the predictions of multiple models. It is also known as bootstrap aggregation. In bagging, the base classifiers are trained in parallel [65].

3.2.2. Boosting

By merging multiple homogenous weak learners to reduce training mistakes, boosting seeks to develop a powerful learner. Boosting, as opposed to bagging, chooses a random sample of data and trains models in a sequential manner. Every new model is created to improve on the shortcomings of the one before it, concentrating on situations that were misclassified. Weak rules from each individual classifier are integrated in this iterative process to produce a strong prediction rule. Boosting gradually enhances the performance of the ensemble model by highlighting the difficult cases and modifying the weights of the training data. In boosting they are trained sequentially [66].

3.2.2.1. XGBoost

XGBoost is a decision-tree-based supervised learning algorithm. It works by building weak DT models in a gradient boosting framework sequentially, then merging them. It gradually improves its predictions by creating new trees to correct the flaws in older ones. The regularized objective function used by XGBoost combines a regularization term to reduce complexity with a loss function to evaluate model performance. It effectively computes the gradient and second-order derivative of the loss function during training in order to optimize the objective function. Large-scale dataset management, parallel processing, automated feature selection, and efficient handling of missing values are just a few of

XGBoost's primary characteristics, which make it a popular choice for a variety of ML applications [67].

3.2.3. AdaBoost

AdaBoost is a supervised ML algorithm that commonly uses DTs with only one level, also known as decision stumps, as its base estimator. The algorithm initially assigns equal weights to all data points and then iteratively trains models, giving higher weights to incorrectly classified points. The subsequent models focus on those misclassified points, aiming to reduce the overall error [68].

3.2.4. Hard Voting

A hard voting ensemble classifier predicts the final class label by counting the majority votes given from the base classifiers. It is also known as majority voting. The idea behind maximum voting is to collect predictions for each class label and then base a forecast on the class label that received the most votes. For instance, when three classifiers (C1, C2, and C3) are combined and they classify a training sample with the classifications [0, 0, 1], applying the mode function [0, 0, 1] results in y = mode [0, 0, 1] = 0. The sample would then be labeled as being of "class 0." In order to make a more certain and accurate forecast for classification problems, this approach takes advantage of the consensus among many classifiers [69]. The idea of using hard voting technique in the proposed work is to reduce overfitting by combining weak learners to get a more reliable outcome.

3.2.5. Soft Voting

A soft voting ensemble classifier predicts the final class label with the highest sum probability by adding the predicted probabilities for the various class labels. It is also known as weighted voting. It is a more complex method when the decision-making process takes into account the actual confidence probabilities of each forecast. In contrast to hard voting, which uses the simple majority rule, soft voting takes into consideration each classifier's prediction confidence. The technique is based on averaging across all models the confidence probabilities associated with each anticipated class label. The ultimate forecast is then made using this averaged data [70]. The idea of using soft voting in the proposed work is to make use of weighted probabilities among the classifiers instead of just relying on majority votes.

3.2.6. Stacking

In stacking, meta-learner is trained to aggregate predictions from various heterogeneous weak learners. The weak learners are trained in parallel, and the predictions they make serve as the meta-learner's input features. An initial set of inner models are trained using the dataset as part of the multi-stage stacking process, and each model generates an initial prediction. After that, the meta-learner discovers the ideal way to integrate these input predictions to produce a better output prediction. The base models' predictions on out-of-sample data are utilized to train the meta-learner. This implies that the base models are fed data that was not used to train them, and that the predictions made by the base models, along with the anticipated outcomes, serve as the input and output pairs for training the meta-learner [71].

3.2.7. Reasons for choosing EL Techniques

One of the main reasons for choosing EL over other techniques it because of its improved accuracy and greater reliability of the results. Rather than relying on one individual model, it's better to combine results from several models to reduce the risk of getting incorrect outputs from one model. Some ensemble approaches like bagging reduce the variance of the error. The AdaBoost reduces both the bias and the variance parts of the error.

3.2.8. Evaluation Measures

Evaluation measures are the metrics to find the performance of the model. The following evaluation measures are used in our multi class classification problem. As this is a multiclass classification problem, two kinds of evaluation measures are needed. One that shows the overall performance and the other for finding the performance on each individual class.

3.2.9. Overall accuracy

Overall accuracy means the number of correct predictions out of the total predictions made by the model on all classes combined. Equation 1 shows the formula for calculating the overall accuracy of the model.

$$Accuracy = \frac{(Number\ of\ correct\ predictions)}{(Total\ number\ of\ Predictions)} \qquad (1)$$

3.2.10. Accuracy for Individual Class

Accuracy for individual class is calculated by dividing the total number of correct predictions on a specific class by the

total number of predictions. Equation 2 shows the formula for calculating the accuracy for individual classes.

$$Accuracy\ for\ Individual\ class = \frac{(Number\ of\ correct\ predictions\ on\ a\ specific\ class)}{Total\ number\ of\ predictions\ on\ a\ specific\ class} \quad (2)$$

3.2.11. Macro Average precision

The macro-average precision (MAP) is the arithmetic mean of individual classes' precision. Equation 3 shows the formula for calculating the MAP.

$$MAP = \frac{(Precision\ on\ class\ 0 + Precition\ on\ class\ 1 + ..+ Precision\ on\ class\ N)}{N} \quad (3)$$

where N is the total number of classes.

3.2.12. Precision for individual class

Precision for a specific class is the ability to correctly classify a specific instance of a class belonging to that class. Equation 4 shows the formula for calculating the precision for individual classes.

$$Precision = \frac{(True\ positives\ for\ the\ class)}{Total\ predicted\ positives\ for\ the\ class} \quad (4)$$

3.2.13. Macro average recall

The macro-average recall (MAR) is the arithmetic mean of individual classes' recall. Equation 5 shows the formula for calculating the MAR.

$$MAR = \frac{(Recall\ on\ class\ 0 + Recall\ on\ class\ 1 + \cdots + Recall\ on\ class\ N)}{N} \quad (5)$$

where N is the total number of classes.

3.2.14. Recall for individual class.

Recall is the number of instances correctly classified as belonging to a specific class out of all instances that actually belong to that class. It is also known as sensitivity. Equation 6 shows the formula for calculating the recall for individual classes.

$$Recall = \frac{(True\ positive\ for\ the\ class)}{(Total\ actual\ positives\ for\ the\ class)} \quad (6)$$

3.2.15. Macro average F1-Score

The macro-average F1-Score (MAF) is the arithmetic mean of individual classes' F1-Score. Equation 7 shows the formula for calculating the MAF.

$$. MAF = \frac{(F1Score\ on\ class\ 0 + F1Score\ on\ class\ 1 + \cdots + F1Score\ on\ class\ N)}{N} \quad (7)$$

where N is the total number of classes.

3.2.16. F1-Score for individual class

By considering both precision and recall, F1-Score provides a balanced means of evaluation. Equation 8 shows the formula for calculating the F1Score for individual classes.

$$F1\ Score = 2 * \frac{(Precision * Recall)}{(Precision + Recall)} \quad (8)$$

3.2.17. Time Elapsed

It represents the time taken from the start of the prediction process until the completion of all predictions or inferences. It is a helpful measure especially in time sensitive applications like detecting attacks in network to mitigate them timely. Equation 9 shows the formula for calculating the elapsed time.

$$\text{Time Elapsed} = \text{End time} = \text{Start time} \quad (9)$$

3.2.18 False Positive and False Negative

False positives and false negatives are crucial metrics for assessing the effectiveness of a system, particularly in situations like as intrusion detection, where accurately distinguishing between normal behaviour and recognizing attacks is of paramount significance. Equations (10) and (11) represents the mathematical formulae for False positive and False negative, respectively.

$$False\ Positive\ (FP) = \frac{Number\ of\ attaks\ incorrectly\ identified\ as\ attacks}{Total\ number\ of\ actual\ negative\ (TN + FP)} \quad (10)$$

$$False\ Negative\ (FN) = \frac{Number\ of\ attaks\ incorrectly\ classified\ as\ negative}{Total\ number\ of\ actual\ positive\ (TP + FN)} \quad (11)$$

4. Methodology

This chapter describes the systematic approach to implementing the proposed model step by step. This section provides a detailed account of dataset description, data preprocessing techniques, implementation the models, process flow, etc. contributing to the existing body of knowledge in the field of IIoT in edge computing.

The following figure 1 represents the process flow of the proposed model.

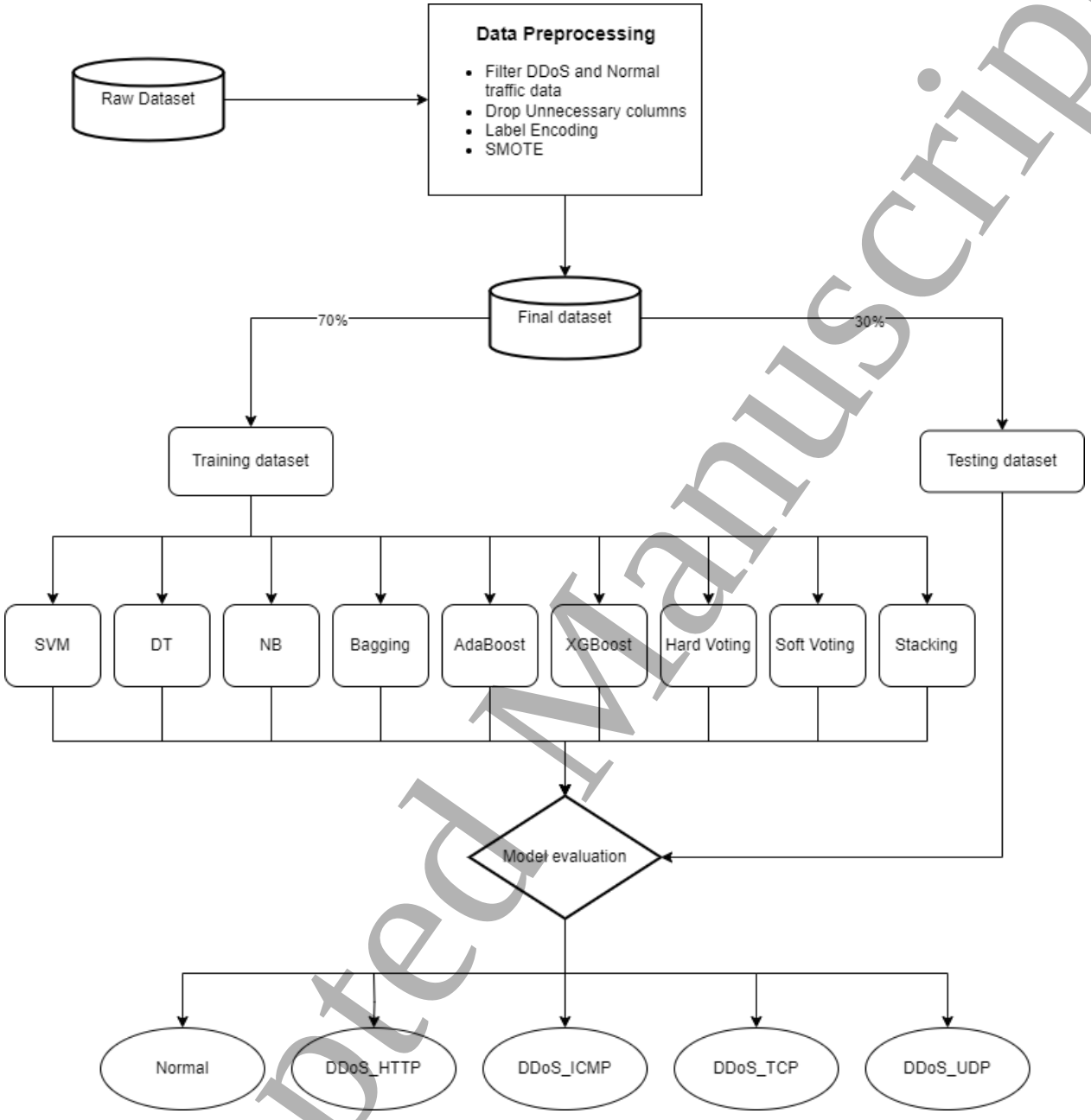

Figure 1 Process flow of the proposed model.

4.1.1. Dataset Description

The novel dataset used in this research is Edge-IIoT, publicly available on Kaggle. This dataset is a "Document in the top 1% of Web of Science." In terms of citations and influence, it is listed in the top 1% of all articles included in the Web of Science [72]. It is a new dataset that contains the most recent kinds of attacks. The data is generated from more than 10 IoT

devices. The original dataset contains the records for 5 categories of attacks and 14 types of attacks. However, our research only focuses on DDoS attacks, so we filter out all other types of attacks and left with only DDoS attack data and Normal traffic data from the entire dataset. The DDoS attacks have four further types in the dataset named TCP SYN flood attack, UDP flood attack, HTTP flood attack, and ICMP flood attack. The dataset has two class labels. One is "Attack Label" and the other is "Attack Type". The former contains binary values, indicating whether there is an attack or not. The later contains the label, indicating the name of the attack type in case of an attack or "Normal" in case of no attack. The following table 2 describes the count of the total selected observations.

Table 2: Count of the Total Selected Observations

Class	Total
Normal	24301
DDoS_HTTP attack	10561
DDoS_ICMP attack	14090
DDoS_TCP attack	10247
DDoS_UDP attack	14498
Total Records	73697

The above table shows the number of selected records for DDoS attack and normal traffic after filtering all other attack types from the raw dataset.

4.1.2. Computational Environment and Configuration

In this study, the computational analysis is conducted using Google Colab, a cloud-based platform offering Python 3 backend support. Regarding the resources, the Google Colab provided Google Compute Engine, which is an Infrastructure as a Service (IaaS) and the environment comprised of 12.67 GB of RAM and 107.72 GB of disk space to run the notebook.

4.1.3. Data Preprocessing

In order for the data to be easily understood and processed by the ML model, as well as to improve accuracy and reduce the time it takes to obtain results from the model, the data must first be cleaned before applying any ML algorithm. As the real-world's data is noisy, contains missing values, in a different format, contains duplicates, and is often unbalanced, it is necessary to convert the raw data into a suitable format for the model training.

Below are some preprocessing techniques that are applied in the proposed model.

4.1.4. Filter All Other Attack Types Apart from DDoS and Normal Traffic

We filtered out all other traffic data apart from DDoS and Normal because our research is focused only on DDoS attack detection. For that, we have used a straightforward approach by first loading the entire dataset into pandas data frame that contained all the attack types. Then we selected only the records for normal traffic and DDoS traffic. So, the final dataset is left with the records of only five classes i-e normal, DDoS HTTP attack, DDoS ICMP attack, DDoS TCP attack, DDoS UDP attack, and normal traffic.

4.1.5. Drop Unnecessary Columns

Out of 63 columns, we dropped 15 unnecessary columns that did not play a major role in identifying the traffic characteristics (the authors of the dataset also dropped these columns before applying the algorithms). We drop the columns like frame.time, ip.src host, ip.dst host, arp.src.proto ipv4, arp.dst.proto ipv4, http.file data, http.request.full uri, icmp.transmit timestamp, http.request.uri.query, tcp.options, tcp.payload, tcp.srcport, tcp.dstport, udp.port, mqtt.msg and attack label. Table 3 lists all the remaining 47 features of the dataset we utilized.

Table 3: Selected features of the dataset

Feature Name
arp.opcode
arp.hw.size
icmp.checksum
icmp.seq_le
icmp.unused
http.content_length
http.request.method
http.referer
http.request.version
http.response
http.tls_port
tcp.ack
tcp.ack_raw
tcp.checksum
tcp.connection.fin
tcp.connection.rst
tcp.connection.syn
tcp.connection.synack
tcp.flags
tcp.flags.ack

tcp.len
tcp.seq
udp.stream
udp.time_delta
dns.qry.name
dns.qry.name.len
dns.qry.qu
dns.qry.type
dns.retransmission
dns.retransmit_request
dns.retransmit_request_in
mqtt.conack.flags
mqtt.conflag.cleansess
mqtt.conflags
mqtt.hdrflags
mqtt.len
mqtt.msg_decoded_as
mqtt.msgtype
mqtt.proto_len
mqtt.protoname
mqtt.topic
mqtt.topic_len
mqtt.ver
mbtcp.len
mbtcp.trans_id
mbtcp.unit_id
Attack_type

4.1.6. Drop Duplicate Records

Duplicate records were dropped from the dataset to make it clean and memory efficient. The first occurrence of a duplicated row is kept, and all other duplicates were removed.

4.1.7. Label Encoding

As machines always deal with numerical data, however, the real-world data often contains categorical data. That's where label encoding comes. It is the process of converting categorical data into numerical data so that can be easily understood by the model. The dataset contains some categorical values which must be converted to numerical values before feeding the data to the model [73]. The attack label contains categorical data in textual representation like DDoS_HTTP, DDoS_ICMP attack, DDoS_TCP, DDoS_UDP, and Normal. Through label encoding they are converted to numerical form where each numerical value corresponds to a certain attack label. Now the values 0, 1, 2, 3, 4 indicate the class labels DDoS_HTTP, DDoS_ICMP attack, DDoS_TCP, DDoS_UDP, and Normal respectively. For label encoding the LabelEncoder() function is used from the built-in library sklearn.preprocessing. All the columns of the dataframe has been iterated and converted to numerical representation using the function fit_transform() which returns the encoded labels.

4.1.8. Training and Testing Data

The dataset is split using the train test split function, where 70% of the data is allocated for the training phase, and 30% is reserved for the testing phase. The same ratio is used for all ML algorithms. First the data is split into two parts named X and y. The X represents all the data except the last two columns which were the class labels. The y contains the last column that represents the attack label. As the proposed model is based on multi-class classification we considered the attack label as the target variable. Now the dataset X and corresponding target variable y will be split into four separate sets after using train_test_split function. The X_train represses training features, X_test represents testing features, y_train represents training target values, and y_test represents testing target values. The training set is used to train the ML model, while the testing set is used to evaluate the model's performance on unseen data. The random state was set to 42 to ensure the same random splitting of the data occurs each time the code is executed, allowing for consistent and comparable results. The following table 4 shows the count of the selected observations for the training and testing samples for each class.

Table 4: Count of the selected observations for training and testing for each class

Class	Training samples	Testing samples
4 (Normal)	16836	7265
3 (DDoS_UDP)	10157	4341
1 (DDoS_ICMP)	9116	3980
0 (DDoS_HTTP)	7337	3158
2 (DDoS_TCP)	7259	2988

The above table show the selected number of observations for training and testing dataset after splitting the processed dataset.

4.1.9. Data balancing using SMOTE.

In ML, unbalanced data distribution occurs when one class contains less or more observations than the other classes. This

presents a challenge since minority classes are frequently misclassified by typical ML algorithms, which favor the dominant class and neglect the minority class. Recall may suffer as a result of this bias towards the majority class, which can also lead to an unbalanced model's performance overall. Different strategies are used to overcome this problem, and one well-liked option is SMOTE. By generating synthetic instances for the minority class, SMOTE seeks to balance the distribution of classes. By extrapolating between current minority class instances and their closest neighbours, it creates new training records. SMOTE assists in overcoming over fitting concerns brought on by random oversampling by carefully oversampling the minority class [74], [75]. From table III, it is clear that the class distribution is imbalanced. So, to have equal number of samples of each class in the training data it is necessary to balance the data in order for the model to avid biasness, poor generalization, accuracy paradox, and insufficient learning for minority classes. Here, SMOTE is applied to the training data that oversamples the data for minority classes by generating synthetic data that closely resembles the actual data [76]. It can be seen in table 3, that after applying SMOTE, all the minority classes are oversampled to match the occurrences of the majority class instances. Now all the classes have equal number of records in the training dataset. It will help the model to learn equally about each type of traffic data. The following table 5 shows the count of the total observations for training and testing for each class after applying SMOTE.

Table 5: Count of total observations for training and testing for each class after applying SMOTE.

Class	Training samples	Testing samples
4 (Normal)	16836	7265
3 (DDoS_UDP)	16836	4341
1 (DDoS_ICMP)	16836	3980
0 (DDoS_HTTP)	16836	3158
2 (DDoS_TCP)	16836	2988

The above table shows the total number of selected observations for training and testing dataset after data balancing using SMOTE. Now the training dataset contains an equal number of samples for each class.

4.1.10. Model Training

For model's training the resampled training data is fed to individual and the ensemble classifiers using the fit () function after importing the required packages for each algorithm. When the training is complete the models are given the testing data to predict their class label using the predict () function. Using the time library, the start time and the end time of the algorithm is captured, and their difference is calculated which gives the total time consumed by the algorithm from its training till its predictions. We have applied Cross family arbitrary selection of state-of-the-art algorithms. SVM, DT, and NB are some well-known techniques for multi-class classification problems. SVM is well-suited for multi-class classification problems due to its ability to handle high-dimensional data effectively and find complex decision boundaries. DTs are intuitive and easy to interpret, making them suitable for explaining the reasoning behind classification decisions. NB is simple yet effective probabilistic classifiers based on Bayes' theorem and the assumption of independence between features. Then we applied well-known EL techniques on these algorithms to see if they yield better outcomes than individual classifiers. The SVM classifier is applied with the default parameters. The default kernel is utilized, which is the Radial Basis Function (RBF). Also, the penalty parameter C is set to 1.0 which is the default value. The DT classifier has been applied with a maximum depth of 3 to avoid over fitting. The NB's variant Gaussian NB is used because the dataset has some continuous values, and the Gaussian NB is good at handling continuous values. The bagging classifier is used with DT as its base estimator with estimators=2, which means that two base estimators will be selected. The maximum depth of each DT is the same i-e 3. Similarly in boosting ensemble, two main boosting techniques are used, AdaBoost and XGBoost. The AdaBoost is also used with DT as its base estimator with two trees [77]. The XGBoost works by default with DT's. In voting ensemble, two techniques have been used. One is hard voting that counts the majority votes for the class label from each individual classifier and predicts the final class label. In both hard and soft voting an ensemble of SVM, DT, and NB were used because these were also the individual classifiers selected. However, in soft voting there is also another level of classifier known as Meta classifier. LR was used as a meta-classifier. The base classifiers individually generate probability estimates for each class during soft voting. Weighted averaging is used to aggregate these probabilities [78]. Due to its probabilistic character, logistic regression is well equipped to handle these probability

estimations and base a final judgment on them. Similarly, in stacking, the same algorithms SVM, DT, and NB were used as the base classifiers.

The above flowchart is the main model of the research that shows different stages of data processing, model generation, and evaluation in a proper flow.

5. Results and Discussions

From the predictions made by the model, the model evaluation has been done using the metrics accuracy, precision, recall, F1-Score, and elapsed time in seconds. Both overall insights, and the deep insight of each model is calculated. The overall insights include the overall accuracy, macro average precision, recall, and F1-Score. The confusion matrix of each classifier is discussed. As this is a multi-class classification problem, macro average evaluation measures and evaluation measures for each individual class has been computed.

The below table 6 shows the overall insights of each model.

Table 6: Overall evaluation measures for each model

Algorithm	Accuracy	Precision	Recall	F1-Score	Time
SVM	53.74	61.08	58.83	52.08	816.84 sec
DT	66.57	65.99	79.99	69.23	0.177 sec
NB	61.25	77.60	68.57	63.21	0.066 sec
Bagging	**99.99**	**99.99**	**99.99**	**99.99**	**0.317 sec**
AdaBoost	**99.99**	**99.99**	**99.99**	**99.99**	**0.503 sec**
XGBoost	**100**	**99.99**	**99.99**	**99.99**	**13.82 sec**
Hard Voting	65.82	72.27	75.27	65.42	943.21 sec
Soft Voting	61.60	77.75	68.92	63.50	3880.85 sec
Stacking	89.38	90.84	92.89	90.66	3134.86 sec

From the above table, it can be concluded that most of the EL techniques performs better than the individual classifiers. In terms of accuracy the ensemble classifiers outperformed all the individual classifiers, except in case of DT where the hard voting classifier and the DT has almost the same accuracy. Now between all the ensemble classifiers, bagging, AdaBoost, and XGBoost outperformed all other ensemble classifiers by achieving 99.99% in all the evaluation measures. However, in terms of elapsed time, ensemble techniques like hard voting, soft voting, and stacking took much longer time because hard voting and stacking contained three classifiers trained together, which obviously takes more time than individual classifiers. While the soft voting had four classifiers, that took highest time to train and made predictions. So, in terms of elapsed time NB took the lowest time to train. While in the case of ensemble techniques the bagging classifier took the lowest time of 0.317 seconds. So,based on the accuracy, precision, recall, and F1-Score it can be concluded that the three ensemble classifiers, XGBoost, AdaBoost, and bagging classifier has achieved the best results. The following table 7 represents the overall evaluation measures for each model on each individual class.

Table 7: Overall evaluation measures for each model on each individual class.

Algorithm	Metrics	Normal	DDoS_HTTP	DDoST_ICMP	DDoS_TCP	DDoS_UDP
SVM	Accuracy	73.81	80.20	79.63	84.48	89.32

Algorithm	**Metrics**	**Normal**	**DDoS_HTTP**	**DDoST_ICMP**	**DDoS_TCP**	**DDoS_UDP**
	Precision	81.58	29.21	46.71	47.89	100
	Recall	27.99	27.22	100	92.08	46.88
	F1-Score	41.68	28.18	63.67	63.01	63.84
	FP	5231	2256	0	247	2320
	FN	459	2045	4425	3125	0
DT	Accuracy	66.57	99.99	99.99	66.57	99.99
	Precision	0.00	100	100	30.04	99.93
	Recall	0.00	100	99.99	100	100
	F1-Score	0.00	100	99.99	46.21	99.22
	FP	7265	0	1	0	0
	FN	0	0	0	7263	3
NB	Accuracy	71.96	67.18	90.37	93.79	71.96
	Precision	100	26.26	64.85	96.92	100
	Recall	16.13	71.93	100	58.65	16.13
	F1-Score	27.78	38.47	78.68	70.08	27.78
	FP	6093	870	0	1290	169
	FN	0	6262	2102	58	0
Bagging	**Accuracy**	**99.95**	**100**	**99.99**	**100**	**99.99**
	Precision	**100**	**100**	**100**	**100**	**99.74**
	Recall	**99.86**	**100**	**99.97**	**100**	**100**
	F1-Score	**99.93**	**100**	**99.98**	**100**	**99.87**
	FP	**10**	**0**	**1**	**0**	**0**
	FN	**0**	**0**	**0**	**0**	**11**
AdaBoost	**Accuracy**	**100**	**100**	**99.99**	**100**	**99.99**
	Precision	**100**	**100**	**100**	**100**	**99.95**
	Recall	**100**	**100**	**99.97**	**100**	**100**
	F1-Score	**100**	**100**	**99.98**	**100**	**99.97**
	FP	**2**	**0**	**1**	**0**	**0**
	FN	**1**	**0**	**0**	**0**	**2**

Algorithm	**Metrics**	**Normal**	**DDoS_HTTP**	**DDoST_ICMP**	**DDoS_TCP**	**DDoS_UDP**
XGBoost	**Accuracy**	**100**	**100**	**100**	**100**	**99.99**
	Precision	**100**	**100**	**100**	**100**	**99.99**
	Recall	**99.99**	**100**	**100**	**100**	**100**
	F1-Score	**99.99**	**100**	**100**	**100**	**99.99**
	FP	**1**	**0**	**0**	**0**	**0**
	FN	**0**	**0**	**0**	**0**	**1**
Hard Voting	Accuracy	71.54	81.56	90.32	88.98	99.22
	Precision	100	41.55	64.85	57.12	100
	Recall	14.89	71.93	100	93.42	96.13
	F1-Score	25.92	52.67	78.68	70.89	98.02
	FP	6183	870	0	205	169
	FN	0	3137	0	2188	2102
Soft Voting	Accuracy	71.95	67.18	90.69	93.78	99.58
	Precision	100	26.26	65.73	96.82	99.97
	Recall	16.10	71.93	100	58.65	97.93
	F1-Score	27.74	38.47	79.32	73.05	98.94
	FP	6095	870	0	1290	90
	FN	0	6262	2022	60	1
Stacking	Accuracy	89.37	100	99.99	89.39	99.98
	Precision	96.17	100	100	58.12	99.93
	Recall	71.03	100	99.97	93.46	100
	F1-Score	81.71	100	99.98	71.67	99.96
	FP	2168	0	0	183	1
	FN	184	0	0	2162	6

The above table shows a deep insight for each model by the evaluation measure achieved on each individual class. It can be seen that the XGBoost, AdaBoost, and bagging, achieved the best score on each individual class as well. The rest of the models also achieved remarkable scores on some evaluation measures in individual classes. The following figure 2 shows the bar chart representing the performance of SVM.

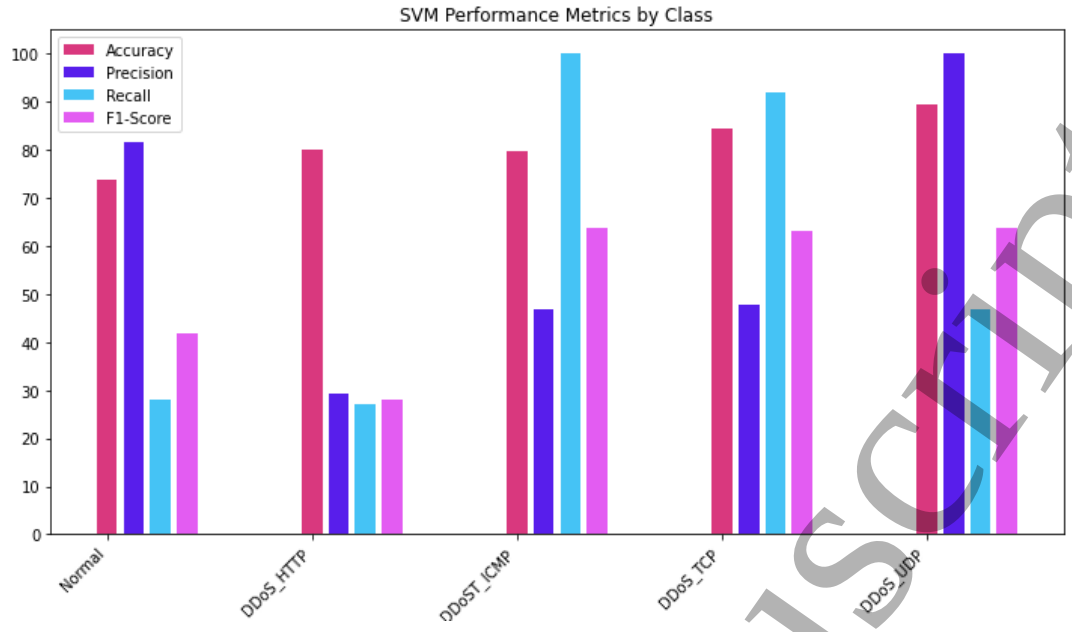

Figure 2 Bar graph of SVM performance metrics

The following figure 3 shows the bar chart representing the performance of DT.

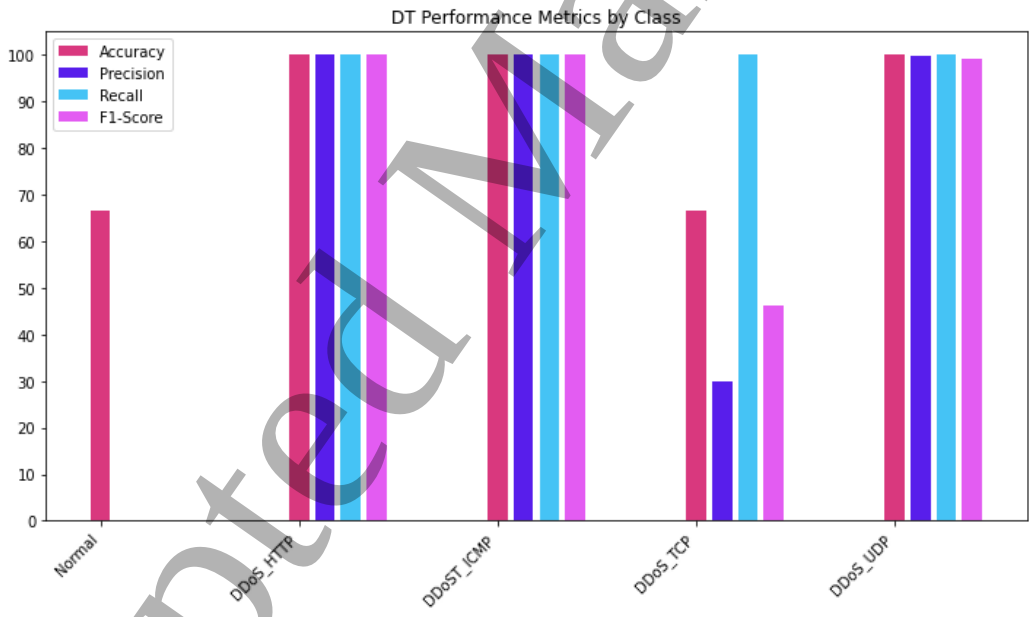

Figure 3 Bar graph for DT performance metrics

The following figure 4 shows the bar chart representing the performance of NB.

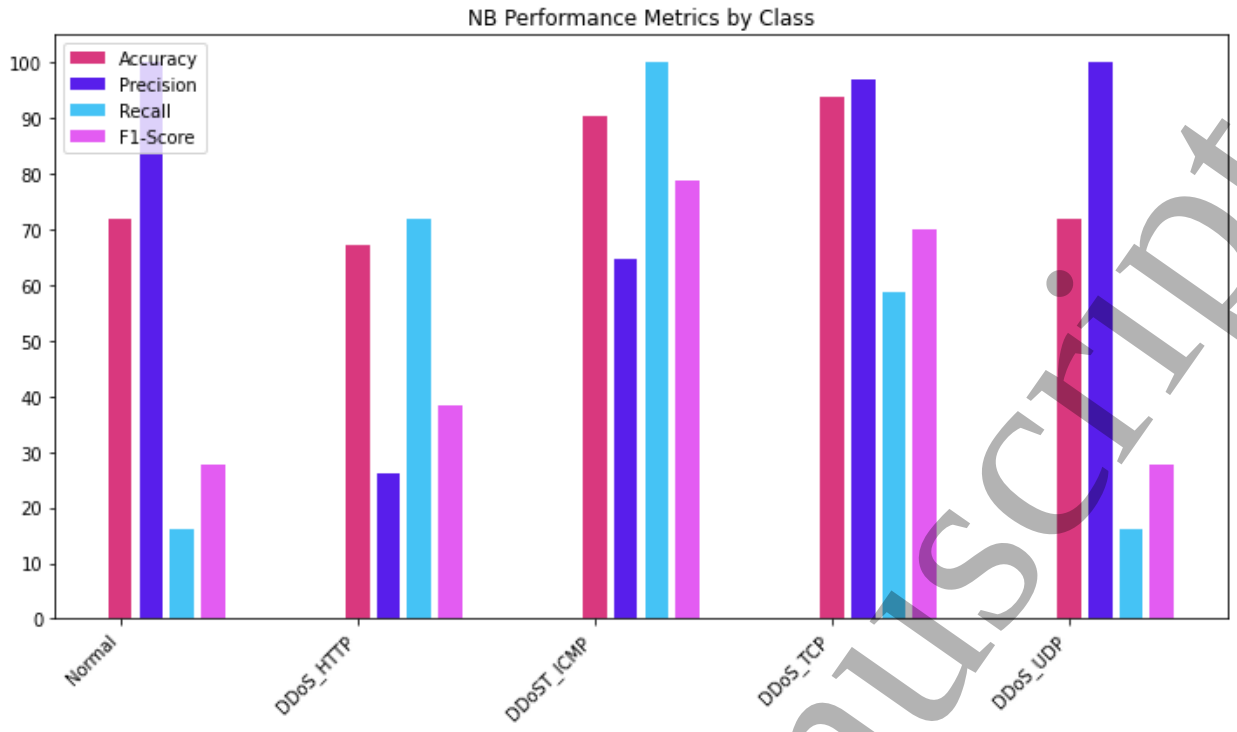

Figure 4 Bar graph for NB performance metrics

The following figure 5 shows the bar chart representing the performance of bagging ensemble.

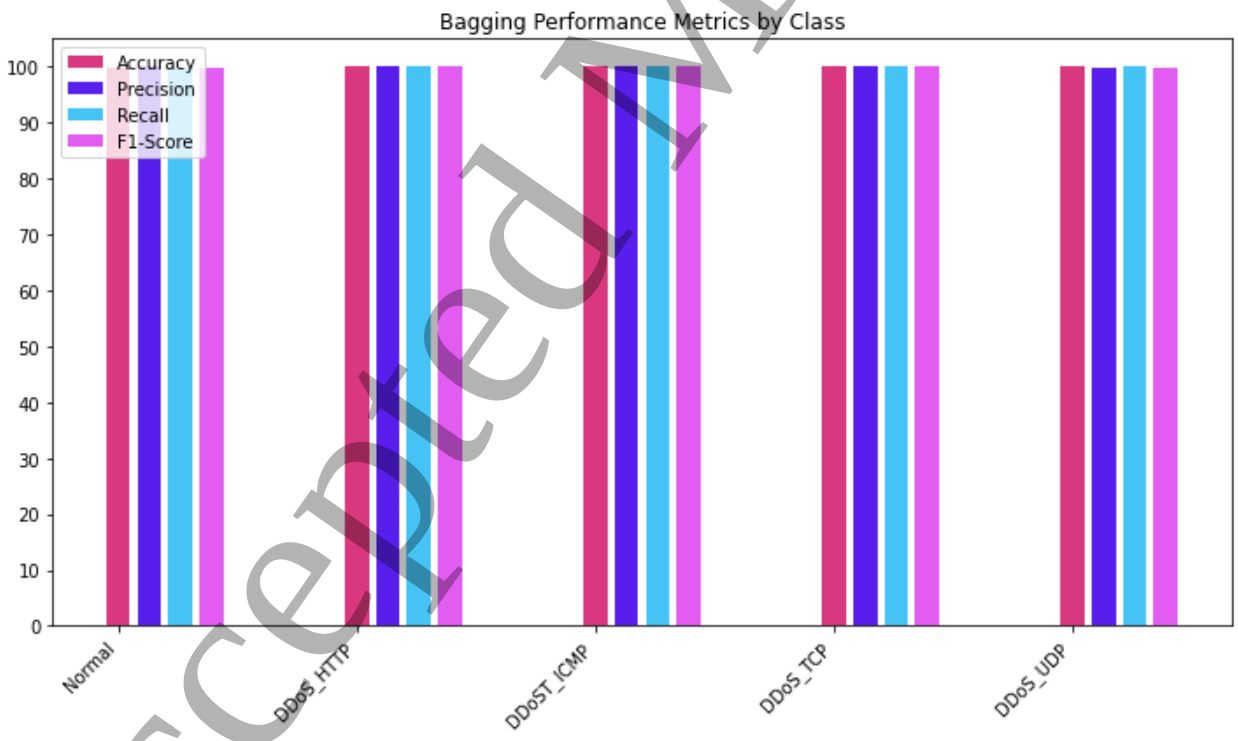

Figure 5 Bar graph for bagging performance metrics

The following figure 6 shows the bar chart representing the performance of AdaBoost ensemble technique.

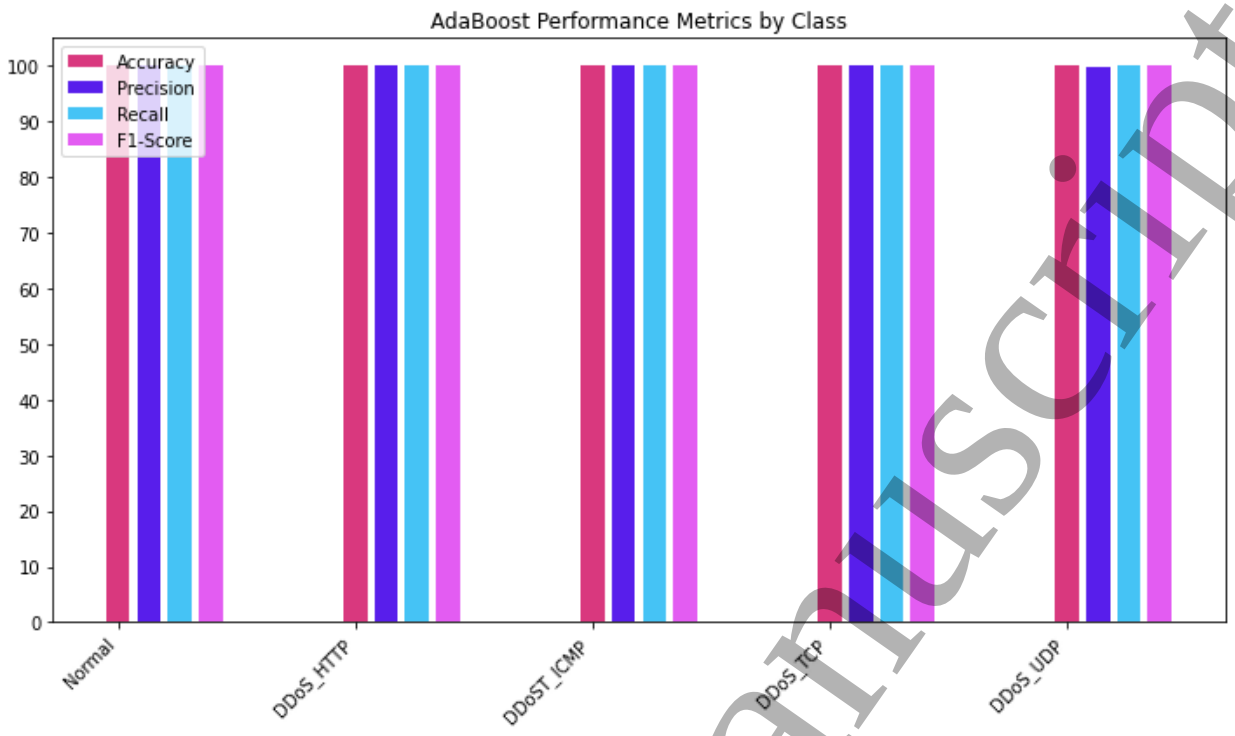

Figure 6 Bar graph for AdaBoost performance metrics

The following figure 7 shows the bar chart representing the performance of XGBoost classifier.

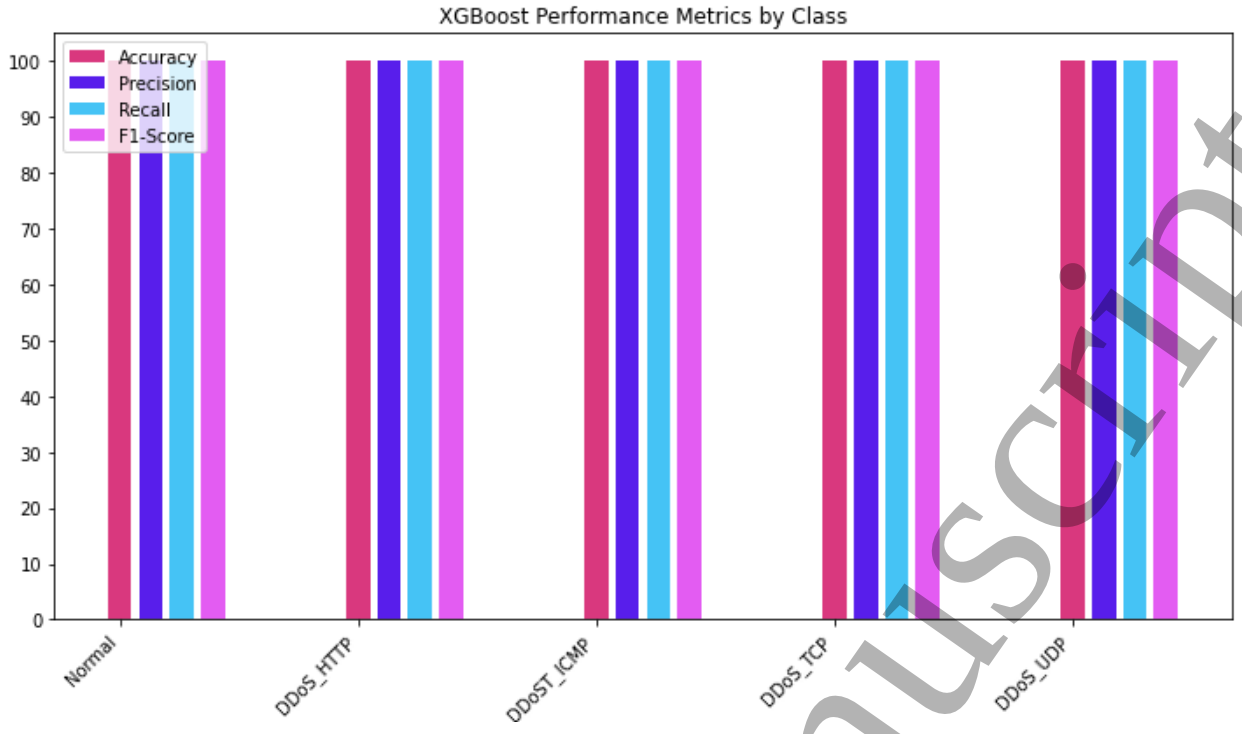

Figure 7 Bar graph for XGBoost performance metrics

The following figure 8 shows the bar chart representing the performance of hard voting ensemble technique.

Figure 8 Bar graph for hard voting performance metrics

The following figure 9 shows the bar chart representing the performance of soft voting ensemble technique.

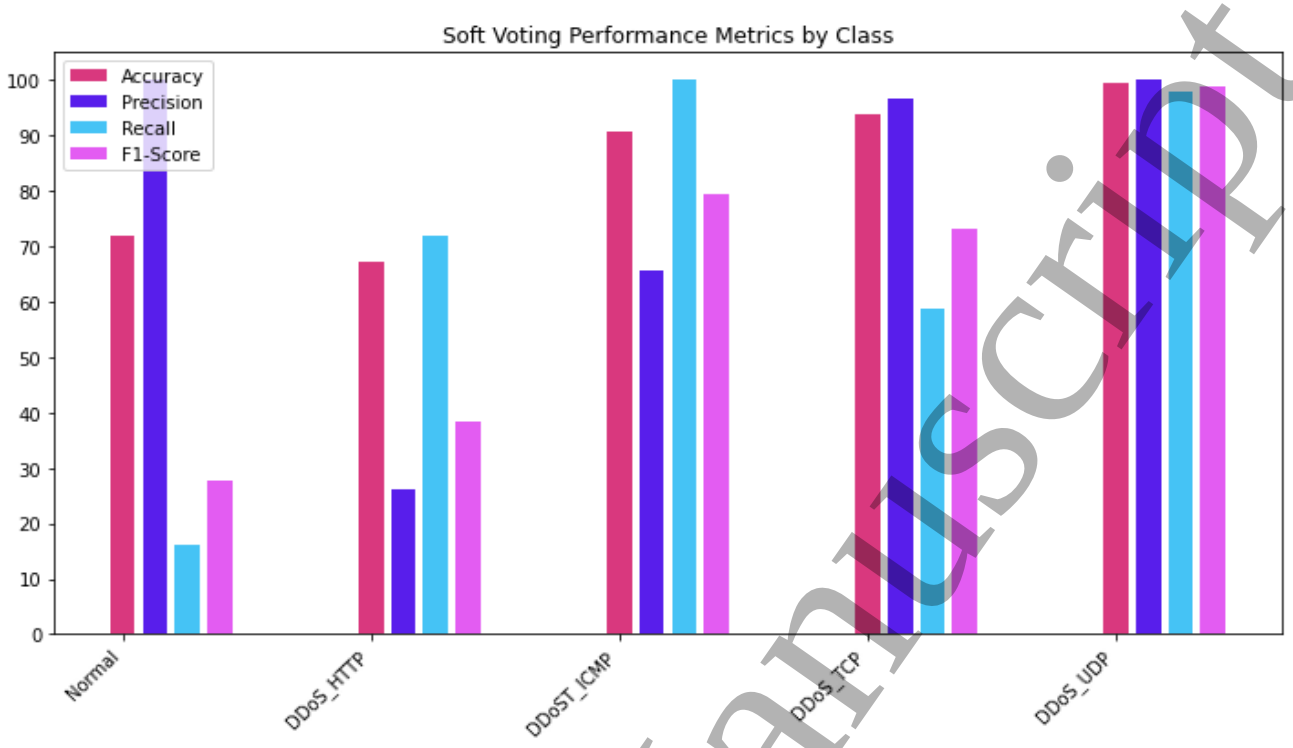

Figure 9 Bar graph for soft voting performance metrics

The following figure 10 shows the bar chart representing the performance of stacking ensemble technique.

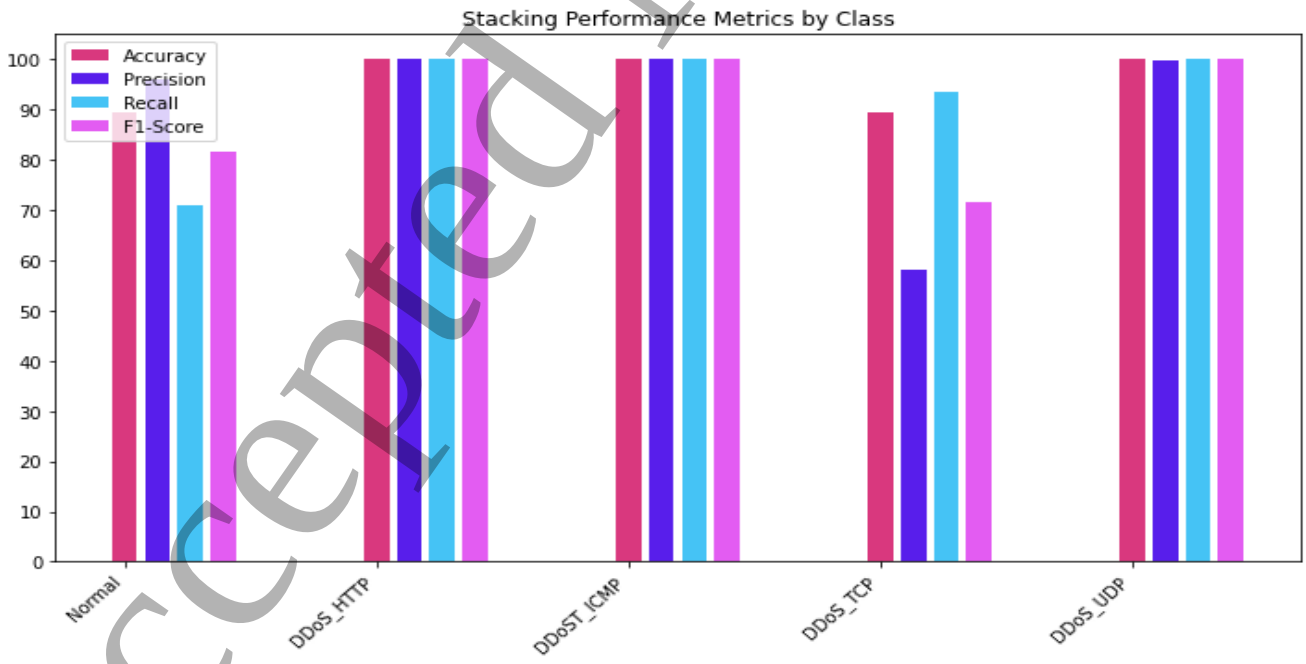

Figure 10 Bar graph for stacking performance metrics.

The following figure 11 shows the bar chart representing the elapsed time of all the algorithms in a sorted order.

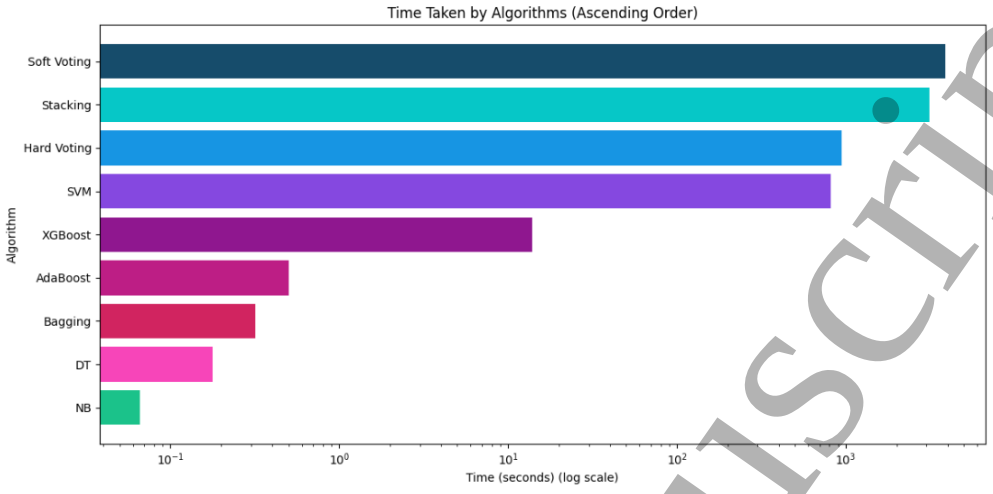

Figure 11 Bar graph depicting the comparison between all classifiers in terms of time taken.

5. Conclusion

Due to the advance growth in IoT technology, it is vulnerable to attacks since everything connected to the internet makes is open to attack. This research detects DDoS attacks along with its type in IIoT network of edge computing and shows deep insights that how EL approaches accurately classifies the traffic type compared to individual classifiers. Through extensive experimentation and evaluation, the research demonstrated that ensemble methods, such as bagging, boosting, stacking, and voting offer significant improvements in the accuracy and robustness of IDS compared to individual classifiers. Furthermore, out of all ensemble classifiers the XGBoost, AdaBoost and bagging classifier has achieved the best results of 99.99% across all evaluation measures. The ensembles effectively leverage the diversity of weak learners to mitigate the limitations of individual models and enhance the overall detection performance. Additionally, the results highlight the importance of preprocessing techniques such as label encoding and data balancing in improving the performance of ensemble methods. However, it is important to note that the effectiveness of any DDoS detection system depends on the specific context, network architecture, and the evolving nature of cyber threats. Regular testing, validation, and refinement are essential components of maintaining a robust defence against DDoS attacks.

6. Future Work

In future advanced EL techniques can be applied to the dataset with feature engineering techniques, and other data balancing techniques. Moreover, the scope of the research can also be extended to detect zero-day attacks and the model can learn the new data continuously without explicit retraining. Real-time data analysis and adaptive models that can dynamically learn and update their detection skills are used in continuous ML approaches for detecting zero-day threats. By continually updating the model's knowledge and adjusting to new attack patterns, continuous ML overcomes this problem. It enables the model to take advantage of fresh data as it becomes available and learn from it, keeping it abreast of new attack methods.

References

[1] E. Bertino, K.-K. R. Choo, D. Georgakopolous, and S. Nepal, "Internet of things (IoT): smart and secure service delivery," ACM Transactions on Internet Technology (TOIT), vol. 16, no. 4, article 22, 2016.

[2] H. F. Atlam and G. B. Wills, "IoT security, privacy, safety, and ethics," In Digital Twin Technologies and Smart Cities, 1st ed., Switzerland, Springer Nature Switzerland AG 2020, chp. 8, pp. 123–149, 2020.

[3] Makhdoom, I.; Abolhasan, M.; Lipman, J.; Liu, R.P.; Ni, W. Anatomy of Threats to the Internet of Things. IEEE Commun. Surv. Tutor. 2018, 21, 1636–1675. [CrossRef].

[4] Devan, P., & Khare, N. (2020). An efficient XGBoost–DNN-based classification model for network intrusion detection system. *Neural Computing and Applications*, *32*(16), 12499–12514. https://doi.org/10.1007/s00521-020-04708-x

[5] Tao, Y.; Yu, S.Y.S. DDoS Attack Detection at Local Area Networks Using Information Theoretical Metrics. In Proceedings of the 2013 12th IEEE International Conference on Trust, Security and Privacy in Computing and Communications, Melbourne, Australia, 16–18 July 2013; pp. 233–240.

[6] E. Alomari, S. Manickam, B. B. Gupta, S. Karuppayah, and R. Alfaris, "Botnet-based Distributed Denial of Service (DDoS) Attacks on Web Servers: Classification and Art," Int. J. Comput. Appl., vol. 49, no. 7, pp. 24–32, 2012.

[7] Kambourakis, G.; Kolias, C.; Stavrou, A. The Mirai botnet and the IoT zombie armies. In Proceedings of the MILCOM 2017 IEEE Military Communications Conference (MILCOM), Baltimore, MD, USA, 23–25 October 2017; pp. 267–272

[8] K. York, Dyn statement on 10/21/2016 DDoS attack, Dyn Blog, 2016, http://dyn.com/blog/dyn-statement-on-10212016-ddosattack/.]

[9] S. Hilton, Dyn analysis summary of Friday October 21 attack, Dyn Blog, 2016, http://dyn.com/blog/dyn-analysis-summary-offriday-october-21-attack/.

[10] O. Yevsieieva and S. M. Helalat, "Analysis of the impact of the slow HTTP DOS and DDOS attacks on the cloud environment," in Proc. 4th International Scientific-Practical Conference Problems of Infocommunications. Science and Technology (PIC S&T). IEEE, 2017, pp. 519–523.

[12] Mighan SN, Kahani M (2020) A novel scalable intrusion detection system based on deep learning. Int J Inf Secur: 1–17

[13] Vigoya, L., Fernandez, D., Carneiro, V., & Nóvoa, F. J. (2021). IoT dataset validation using machine learning techniques for traffic anomaly detection. *Electronics (Switzerland)*, *10*(22). https://doi.org/10.3390/electronics10222857

[14] . Hussain, F.; Hussain, R.; Hassan, S.A.; Hossain, E. Machine Learning in IoT Security: Current Solutions and Future Challenges. IEEE Commun. Surv. Tutor. 2020, 22, 1686–1721. [CrossRef]

[15] . Gao X, Shan C, Hu C, Niu Z, Liu Z (2019) An adaptive ensemble machine learning model for intrusion detection. IEEE Access 7:82512–82521

[16]. Li L, Yu Y, Bai S, Hou Y, Chen X (2017) An effective two-step intrusion detection approach based on binary classification and k-NN. IEEE Access 6:12060–12073

[17] Ahmad I, Basheri M, Iqbal MJ, Rahim A (2018) Performance comparison of support vector machine, random forest, and extreme learning machine for intrusion detection. IEEE Access 6:33789–33795

[18] Farnaaz N, Jabbar MA (2016) Random forest modeling for network intrusion detection system. Procedia Comput Sci 89:213–217

[19] Han S, Xie M, Chen HH, Ling Y (2014) Intrusion detection in cyber-physical systems: techniques and challenges. IEEE Sys J 8(4):1052–1062

[20] Mohamed Amine Ferrag, Othmane Friha et al Edge-IIoTset: A New Comprehensive Realistic Cyber Security Dataset of IoT and IIoT Applications for Centralized and Federated Learning

[21] A. Carvalho, N. O' Mahony, L. Krpalkova, S. Campbell, J. Walsh, and P. Doody, "Edge computing applied to industrial machines," *Procedia Manuf.*, vol. 38, no. 2019, pp. 178–185, 2019, doi: 10.1016/j.promfg.2020.01.024.

[22] Y. Xiao, Y. Jia, C. Liu, X. Cheng, J. Yu, and W. Lv, "Edge Computing Security: State of the Art and Challenges," *Proc. IEEE*, pp. 1–23, 2019, doi: 10.1109/JPROC.2019.2918437.

[23] E. Alomari, S. Manickam, B. B. Gupta, S. Karuppayah, and R. Alfaris, "Botnet-based Distributed Denial of Service (DDoS) Attacks on Web Servers: Classification and Art," *Int. J. Comput. Appl.*, vol. 49, no. 7, pp. 24–32, 2012, doi: 10.5120/7640-0724.

[24] *Radware Full Year 2022 Report: Malicious DDoS Attacks Rise 150%*. (n.d.). Retrieved August 1, 2023, from https://www.radware.com/newsevents/pressreleases/2023/radware-full-year-2022-report-malicious-ddos-attacks/

[25] Chartuni, A., & Márquez, J. (2021). Multi-classifier of DDoS attacks in computer networks built on neural networks. *Applied Sciences (Switzerland)*, *11*(22). https://doi.org/10.3390/app112210609

[26] Elsayed, M. S., Le-Khac, N. A., Dev, S., & Jurcut, A. D. (2020). DDoSNet: A Deep-Learning Model for Detecting Network Attacks. *Proceedings - 21st IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks, WoWMoM 2020*, 391–396. https://doi.org/10.1109/WoWMoM49955.2020.00072

[27] Gurulakshmi, K., & Nesarani, A. (2018). Analysis of IoT Bots against DDOS attack using Machine learning algorithm. *2018 2nd International Conference on Trends in Electronics and Informatics (ICOEI)*, *Icoei*, 1052–1057.

[28] Mohammed, S. (2021). A Machine Learning-Based Intrusion Detection of DDoS Attack on IoT Devices. *International Journal of Advanced Trends in Computer Science and Engineering*, *10*(4), 2792–2797. https://doi.org/10.30534/ijatcse/2021/221042021

[29] Apostol, I., Preda, M., Nila, C., & Bica, I. (2021). Iot botnet anomaly detection using unsupervised deep learning. *Electronics (Switzerland)*, *10*(16). https://doi.org/10.3390/electronics10161876

[30] Larriva-Novo, X.; Villagrá, V.A.; Vega-Barbas, M.; Rivera, D.; Sanz Rodrigo, M. An IoT-Focused Intrusion Detection System Approach Based on Preprocessing Characterization for Cybersecurity Datasets. *Sensors* **2021**, *21*, 656. https://doi.org/10.3390/ s21020656

[31] Su, J., He, S., & Wu, Y. (2022). Features selection and prediction for IoT attacks. *High-Confidence Computing*, *2*(2), 100047. https://doi.org/10.1016/j.hcc.2021.100047

[32] Ullah, I.; Ullah, A.; Sajjad, M. Towards a Hybrid Deep Learning Model for Anomalous Activities Detection in Internet of Things Networks. *IoT* **2021**, *2*, 428–448. https://doi.org/10.3390/iot2030022

[33] Palla, T.G.; Tayeb, S. Intelligent Mirai Malware Detection for IoT Nodes. *Electronics* **2021**, *10*, 1241. https://doi.org/10.3390/ electronics10111241

[34] M. Esmaeili, S. H. Goki, B. H. K. Masjidi, M. Sameh, H. Gharagozlou, and A. S. Mohammed, "ML-DDoSnet: IoT Intrusion Detection Based on Denial-of-Service Attacks Using Machine Learning Methods and NSL-KDD," *Wirel. Commun. Mob. Comput.*, vol. 2022, 2022, doi: 10.1155/2022/8481452.

[35] M. M. Rashid, S. U. Khan, F. Eusufzai, M. A. Redwan, S. R. Sabuj, and M. Elsharief, "A Federated Learning-Based Approach for Improving Intrusion Detection in Industrial Internet of Things Networks," *Network*, vol. 3, no. 1, pp. 158–179, 2023, doi: 10.3390/network3010008.

[36] M. H. Ali *et al.*, "Threat Analysis and Distributed Denial of Service (DDoS) Attack Recognition in the Internet of Things (IoT)," *Electron.*, vol. 11, no. 3, 2022, doi: 10.3390/electronics11030494.

[37] K. O. A. Alimi, K. Ouahada, A. M. Abu-Mahfouz, S. Rimer, and O. A. Alimi, "Refined LSTM Based Intrusion Detection for Denial-of-Service Attack in Internet of Things," *J. Sens. Actuator Networks*, vol. 11, no. 3, 2022, doi: 10.3390/jsan11030032.

[38] J. G. Almaraz-Rivera, J. A. Perez-Diaz, and J. A. Cantoral-Ceballos, "Transport and Application Layer DDoS Attacks Detection to IoT Devices by Using Machine Learning and Deep Learning Models," *Sensors*, vol. 22, no. 9, 2022, doi: 10.3390/s22093367.

[39] A. Khacha, R. Saadouni, Y. Harbi, and Z. Aliouat, "Hybrid Deep Learning-based Intrusion Detection System for Industrial Internet of Things," *ISIA 2022 - Int. Symp. Informatics its Appl. Proc.*, no. April 2023, 2022, doi: 10.1109/ISIA55826.2022.9993487.

[40] V. Hnamte and J. Hussain, "DCNNBiLSTM: An Efficient Hybrid Deep Learning-Based Intrusion Detection System," *Telemat. Informatics Reports*, vol. 10, no. March, p. 100053, 2023, doi: 10.1016/j.teler.2023.100053.

[41] F. Ullah, S. Ullah, G. Srivastava, and J. C.-W. Lin, "IDS-INT: Intrusion detection system using transformer-based transfer learning for imbalanced network traffic," *Digit. Commun. Networks*, 2023, doi: 10.1016/j.dcan.2023.03.008.

[42] F. B. Saghezchi, G. Mantas, M. A. Violas, A. M. de Oliveira Duarte, and J. Rodriguez, "Machine Learning for DDoS Attack Detection in Industry 4.0 CPPSs," *Electron.*, vol. 11, no. 4, pp. 1–14, 2022, doi: 10.3390/electronics11040602.

[43] H. C. Altunay and Z. Albayrak, "A hybrid CNN + LSTMbased intrusion detection system for industrial IoT networks," *Eng. Sci. Technol. an Int. J.*, vol. 38, p. 101322, 2023, doi: 10.1016/j.jestch.2022.101322.

[44] S. Attack, M. A. Ferrag, L. Shu, and H. Djallel, "Deep Learning-Based Intrusion Detection for Distributed," pp. 1–26, 2021.

[45] S. Latif, Z. Zou, Z. Idrees, and J. Ahmad, "A Novel Attack Detection Scheme for the Industrial Internet of Things Using a Lightweight Random Neural Network," *IEEE Access*, vol. 8, pp. 89337–89350, 2020, doi: 10.1109/ACCESS.2020.2994079.

[46] W. Yao, L. Hu, Y. Hou, and X. Li, "A Lightweight Intelligent Network Intrusion Detection System Using One-Class Autoencoder and Ensemble Learning for IoT," *Sensors*, vol. 23, no. 8, pp. 1–25, 2023, doi: 10.3390/s23084141.

[47] Dutta, V., Choraś, M., Pawlicki, M., & Kozik, R. (2020). A deep learning ensemble for network anomaly and cyber-attack detection. *Sensors (Switzerland)*, *20*(16), 1–20. https://doi.org/10.3390/s20164583

[48] Yan Song, Haowei Li , Panfeng Xu, and D. L. (2022). A Method of Intrusion Detection Based on WOA-XGBoost Algorithm. *Discrete Dynamics in Nature and Society*, *2022*, 9. https://doi.org/10.1155/2022/5245622

[49] Devan, P., & Khare, N. (2020). An efficient XGBoost–DNN-based classification model for network intrusion detection system. *Neural Computing and Applications*, *32*(16), 12499–12514. https://doi.org/10.1007/s00521-020-04708-x

[50] Ikram, S. T., Cherukuri, A. K., Poorva, B., Ushasree, P. S., Zhang, Y., Liu, X., & Li, G. (2021). Anomaly Detection Using XGBoost Ensemble of Deep Neural Network Models. *Cybernetics and Information Technologies*, *21*(3), 175–188. https://doi.org/10.2478/cait-2021-0037

[51] Sweta, B., Siva, R. K. S., Praveen, K. M., Rajesh, K., Saurabh, S., Thippa, R. G., Mamoun, A., & Tariq, U. (2020). A Novel PCA-Firefly Based XGBoost Classification Model for Intrusion Detection in Networks. *Electronics (Switzerland)*, *9*(2), 219.

[52] Jiang, H., He, Z., Ye, G., & Zhang, H. (2020). Network Intrusion Detection Based on PSO-Xgboost Model. *IEEE Access*, *8*, 58392–58401. https://doi.org/10.1109/ACCESS.2020.2982418

[53] T. T. H. Le, Y. E. Oktian, and H. Kim, "XGBoost for Imbalanced Multiclass Classification-Based Industrial Internet of Things Intrusion Detection Systems," *Sustain.*, vol. 14, no. 14, pp. 1–21, 2022, doi: 10.3390/su14148707.

[54] S. H. Javed, M. Bin Ahmad, M. Asif, S. H. Almotiri, K. Masood, and M. A. Al Ghamdi, "An Intelligent System to Detect Advanced Persistent Threats in Industrial Internet of Things (I-IoT)," *Electron.*, vol. 11, no. 5, pp. 1–25, 2022, doi: 10.3390/electronics11050742.

[55] J. B. Awotunde *et al.*, "An Ensemble Tree-Based Model for Intrusion Detection in Industrial Internet of Things Networks," *Appl. Sci.*, vol. 13, no. 4, 2023, doi: 10.3390/app13042479.

[56] M. Alduailij, Q. W. Khan, M. Tahir, M. Sardaraz, M. Alduailij, and F. Malik, "Machine-Learning-Based DDoS Attack Detection Using Mutual Information and Random Forest Feature Importance Method," *Symmetry (Basel).*, vol. 14, no. 6, pp. 1–15, 2022, doi: 10.3390/sym14061095.

[57] A. Al-Abassi, H. Karimipour, A. Dehghantanha, and R. M. Parizi, "An ensemble deep learning-based cyber-attack detection in industrial control system," *IEEE Access*, vol. 8, pp. 83965–83973, 2020, doi: 10.1109/ACCESS.2020.2992249.

[58] V. Priya, I. S. Thaseen, T. R. Gadekallu, M. K. Aboudaif, and E. A. Nasr, "Robust attack detection approach for iiot using ensemble classifier," *Comput. Mater. Contin.*, vol. 66, no. 3, pp. 2457–2470, 2020, doi: 10.32604/cmc.2021.013852.

[59] B. Thiyam and S. Dey, "Efficient Feature Evaluation Approach for a class-imbalanced dataset using Machine learning," *Procedia Comput. Sci.*, vol. 218, pp. 2520–2532, 2023, doi: 10.1016/j.procs.2023.01.226.

[60] Mishra, J.P., Singh, K. and Chaudhary, H., 2023. Analyzing the effectiveness of MEMS sensor and IoT in predicting wave height using machine learning models. Measurement Science and Technology, 34(7), p.075904.

[61] C. Cortes and V. Vapnik, "Support-vector networks," Machine Learning, vol. 20, no. 3, pp. 273–297, 1995.

[62] Breiman, L., Friedman, J. H., Olshen, R. A., and Stone, C. J., Classification and Regression Trees, Wadsworth, 1984.

[63] Rish, Irina. (2001). An Empirical Study of the Naïve Bayes Classifier. IJCAI 2001 Work Empir Methods Artif Intell. 3.

[64] X. Dong and Z. Yu, “A survey on ensemble learning,” vol. 14, no. 2, pp. 241–258, 2020.

[65] L. E. O. Bbeiman, “Bagging Predictors,” vol. 140, pp. 123–140, 1996.

[66] G. Ridgeway, "The state of boosting," in Computing Science and Statistics, pp. 172-181, 1999.

[67] Chen, T., and Guestrin, C. "Xgboost: A scalable tree boosting system." In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 785-794. August 2016.

[68] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," Journal of Computer and System Sciences, vol. 55, no. 1, pp. 119-139, 1997.

[69] A. Mohammed and R. Kora, “A comprehensive review on ensemble deep learning: Opportunities and challenges,” J. King Saud Univ. - Comput. Inf. Sci., vol. 35, no. 2, pp. 757–774, 2023, doi: 10.1016/j.jksuci.2023.01.014.

[70] N. Lower and F. Zhan, “A Study of Ensemble Methods for Cyber Security,” 2020 10th Annu. Comput. Commun. Work. Conf. CCWC 2020, pp. 1001–1009, 2020, doi: 10.1109/CCWC47524.2020.9031256.

[71] D. H. Wolpert, “Stacked generalization,” Neural Networks, vol. 5, no. 2, pp. 241–259, 1992, doi: 10.1016/S0893-6080(05)80023-1.

[72] DatasetEdge-IIoT - Cyber Security Dataset of IoT & IIoT" [Online]. Available: https://www.kaggle.com/datasets/mohamedamineferrag/edgeiiotset-cyber-security-dataset-of-iot-iiot. [Accessed: September 2, 2023, 5:00 PM].

[73] Neha Sharma et al., “Optimization of IDS using Filter-Based Feature Selection and Machine Learning Algorithms,” Int. J. Innov. Technol. Explor. Eng., vol. 10, no. 2, pp. 96–102, 2020, doi: 10.35940/ijitee.b8278.1210220.

[74] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: Synthetic Minority Over-sampling Technique," Journal of Artificial Intelligence Research, vol. 16, pp. 321-357, 2002.

[75] Amin, A., Anwar, S., Adnan, A., Nawaz, M., Howard, N., Qadir, J., Hawalah, A. and Hussain, A., 2016. Comparing oversampling techniques to handle the class imbalance problem: A customer churn prediction case study. IEEE Access, 4, pp.7940-7957.

[76] Amin, A., Rahim, F., Ali, I., Khan, C. and Anwar, S., 2015. A comparison of two oversampling techniques (smote vs mtdf) for handling class imbalance problem: A case study of customer churn prediction. In New Contributions in Information Systems and Technologies: Volume 1 (pp. 215-225). Springer International Publishing.

[77] Laiq, F., Al-Obeidat, F., Amin, A. and Moreira, F., 2023, October. DDoS Attack Detection in Edge-IIoT using Ensemble Learning. In 2023 7th Cyber Security in Networking Conference (CSNet) (pp. 204-207). IEEE.

[78] Amin, A., Anwar, S., Adnan, A., Khan, M.A. and Iqbal, Z., 2015, November. Classification of cyber-attacks based on rough set theory. In 2015 First International Conference on Anti-Cybercrime (ICACC) (pp. 1-6). IEEE.