Turkish Journal of Engineering

https://dergipark.org.tr/en/pub/tuje

e-ISSN 2587-1366

Real Time Intrusion Detection In Edge Computing Using Machine Learning Techniques

Abhay Pratap Singh Bhadauria*[1], Mahendra Singh [2], Rajesh Kumar [3], Amit Kumar[4]

[1]*Department of Computer Engineerieng & Applications, GLA University Mathura, India, abhaypratap00@gmail.com*
[2]*Department of Computer Science, Gurukula Kangri (Deemed to be University) Haridwar, India, msa@gkv.ac.in*
[3]*Thapar Institute of Engineerieng & Technology (Deemed to be University), Patiala, India, rakumar@thapar.edu*
[4]*Department of Computer Science, Gurukula Kangri (Deemed to be University) Haridwar, India, 19523002@gkv.ac.in*

Cite this study: Bhadauria, A.P.S., Singh M., Kumar, R., & Kumar, A., (2025). Real Time Intrusion Detection In Edge Computing Using Machine Learning Techniques. Turkish Journal of Engineering, 9 (2), 385-393.

https://doi.org/10.31127/tuje.1516046

Keywords
IoT Devices
Edge Computing
IDS
Machine Learning

Research Article

Received:14.07.2024
Revised:20.08.2024
Accepted:27.08.2024
Published:01.04.2025

Abstract

The proliferation of edge computing has introduced new opportunities for optimizing latency-sensitive and bandwidth-intensive applications by processing the data closer to its source. In addition, this paradigm shift also brings forth unique security challenges, particularly in the realm of intrusion detection. In edge computing environments, where data is processed at the network edge closer to the data source, real-time intrusion detection is crucial to safeguard the security of the system. The attackers are also exploiting the edge network with rapid extension. Conversely, conventional Intrusion Detection Systems (IDS) cannot detect the latest types of attack patterns in high-speed real-time networks due to their complex behavior and low processing capability. This study introduces a novel approach for developing an effective IDS model to handle such threats in a real-time network and explores the design and implementation of a real-time intrusion detection system (IDS) tailored for edge computing environments. The proposed model is found to be methodical and reliable, and employs supervised Machine Learning (ML) techniques. The objective is to precisely recognize and categorize harmful intrusions or malignant activities within the network in real-time. In order to train and test the model, a self created dataset which utilizes both malevolent and benign PCAPs (Packet Capture files) is used in this research study. To determine the usefulness of the IDS model, the random forest, decision tree, extra tree, and K-nearest neighbors were used as classification techniques. The proposed IDS model exhibhits excellent performance based on several factors such as adaptability and scalability. The model also generates higher values for accuracy, detection rate, F-measure, precision, recall, and lower FPR.

1. Introduction

The Internet of Things (IoT) stands for interconnection of the physical things that are implanted with sensors, software, and other technologies with the goal of communicating and transfering data to other devices and systems over the internet [1-5]. Edge computing plays an essential role in upgrading the functionality and efficiency of IoT networks and it also mitigates the latency, preserves the bandwidth and strengthens the information security [6-8]. There are mainly two types of IoT networks: centralized and decentralized. Centralized IoT networks reckons on a single point of control or a middle hub which manages all the connected objects. This hub can be a cloud-based platform or a physical object such as a server. In this type of network, all the devices communicate with the hub, which then processes the information and gives back commands to the devices. These types of IoT networks are mainly utilized in industrial and commercial applications where information security and real-time processing are crucial. Decentralized IoT networks, on the contrary, divide the control among the connected objects themselves. It means that the devices interconnect directly with each other, rather than through a central hub. Decentralized IoT networks are more workable and scalable than centralized ones, and they are valuable for applications where the devices need to perform independently without depending on a central point of control [9-11]. IoT networks can utilize several communication protocols, including Wi-Fi, Bluetooth, Zigbee, LoRaWAN, and cellular networks such as 5G [12-13]. Each protocol has its own advantages and disadvantages in terms of range, power

consumption, data transfer rates, and network capacity, therefore the choice of protocol relies on the explicit necessities of the IoT application. These devices increase the vulnerability space for cyber-attacks. Furthermore, IoT systems or Edge Computing (EC) networks are subject to several security attacks such as DoS and DDoS attacks. These styles of attacks can cause momentous damage to IoT services and EC networks. Therefore, the security of EC or IoT becomes more and more important [14]. IDS functions as a security measure predominantly situated within the network stratum of EC systems. The packets must be analyzed by the IDS of the EC system deployed in the multiple layers of the EC network, and responses to these packets must be generated in real time. Therefore, traditional IDSs are not fully compatible with EC networks. In order to develop a mitigation strategy for EC networks, we need to understand their security vulnerabilities and how to mitigate them. This article focuses on essential aspects that affect the performance of IDS in EC, such as detection accuracy, FPR, and processing time. This study also focuses to construct an efficient and robust IDS that provides higher recall, precision, f1-score, and detection time. Furthermore, the vast majority of related work on IDS uses KDD cup99 or NSL-KDD datasets, which are not updated. In this paper, malicious PCAPs were gathered from the packet total [15], while normal traffic was produced using the Wireshark tool [16-17]. Over 80 statistical features related to network traffic were computed and extracted for both normal and malicious Biflows by utilizing the CICFlowmeter software, which can be accessed openely on the website of the Canadian Institute for Cybersecurity[18]. Subsequently, an analysis was conducted on the dataset created for the purpose of classifying malware categories, which is update with new attacks compared to the KDD cup99 dataset and the NSL-KDD dataset. The proposed approach in the study yielded numerous key observations, which are outlined below:

- ML based techniques are employed to classify the network traffic.
- The model developed in the study was constructed, trained, and evaluated using several classifiers, including RF, DT, k-NN, and ET. These classifiers were used to evaluates the model's performance based on several performance indicators.

The rest of the paper is designed as follows. Section 2 illustrates the existing literature and research relevant to the topic. It delivers a summary of the previous work conducted by other researchers in the area and identifies the gap that this paper aims to fill. Section 3 and 4 describes the methodologies and resources utilized in constructing the model. It outlines the approach adapted to address the research problem and explains the tools, datasets, algorithms, or frameworks employed. Section 5 discusses the experimental results. Finally, section 6 concludes the paper.

2. Related Work

This section examines general idea of the state-of-the-art studies on IDS based systems. It delivers a widespread summary of the existing research and advancements in the field of IDS. The purpose of this section is to establish the current knowledge and understanding of IDS systems, highlighting the gaps and limitations that the proposed approach aims to address.

Loganathan et al. [19] established a practical and dynamic auto-scaling flow processor for the real-time identification of potential threats. This system aims to improve query efficiency, present innovative rules for feature expansion, and remove redundant regulations to mitigate resource utilization. Their results show that PSO outperforms Bayesian optimization in the context of optimizing rules for Complex Event Processing (CEP) via the utilization of a simulated loss function. Rathore et al. [20] introduced a novel real-time Intrusion Detection System (IDS) designed to perform properly in high-speed network environments. The system is divided into four layers: the capturing layer, the filtration and load balancing layer, the processing layer, and the decision-making layer. The capturing layer is responsible for collecting network traffic data, that handles as input for the IDS. The filtration and load balancing layer inspects the captured data and contributes the load evenly across the system for reliable processing. The processing layer is the core module of the system and combines of Hadoop master and data nodes. Hadoop, along with Apache Spark as a other-party tool and MapReduce as the backend programming, is utilized for processing the network traffic. To categorize and inspect intrusions in real-time, a combination of RepTree and J48 algorithms is utilized in the processing layer. These algorithms used the proposed features to categorize network traffic precisely. The performance of the system is evaluated on a Hadoop single node setup with Apache Spark and MapReduce. This research conducted on comparative study to evaluate the proposed IDS system. The evaluation determines that the system shows enhanced accuracy, efficiency, and real-time adaptability in high-speed network environments. The authors determined the superiority of their proposed IDS by comparing the results with alternative methods or existing systems.

The study carried out by J. Lee et al. [21] introduces a lightweight IDS called IMPACT (Impersonation Attack detection using deep auto-encoder and feature abstraction). The system is specifically designed to operate efficiently on resource-constrained devices. IMPACT utilizes a stacked autoencoder (SAE) and mutual information (MI) techniques for feature extraction and selection. These type of method to helped reduce the number of features, making it appropriate for deployment on devices with restricted computational resources. The extracted and choosed features are then used to train a gradient-based linear Support Vector Machine (SVM) and C4.8 wrapper, which is the classification algorithm used in this IDS. The system is trained on the Aegean Wi-Fi Intrusion Dataset (AWID) to recognize impersonation threats. The statistical findings retrieved from the evaluation of IMPACT determines its effectiveness. The presented IDS achieves an accuracy of 98.22%, with a detection rate of 97.64% and a false alarm rate of 1.20. Shaikh et al. [22] presented IDS model that is relyed on deep learning method. This model uses ResNet50, a CNN made up of 50 layers. The profit of ResNet is that it delievers an advancement over tradational ML methods,

which are in-appropriate to mitigate false alarm rates. All network packets are classified into normal or malevolent groups by the started IDS model with respect to identification of intrusions. By employing these three datasets (KDD99, CICIDS2017, and UNSWNB15), the presented model in the simulated work is trained and validated on diverse network traffic data with several attack types and normal activities. This method gurantees that the model's performance is evaluated comprehensively and contributes a robust assessment of its effectiveness in recognizing and categorizing different types of network intrusions. To verify the model's overall accuracy, recall, precision, and F1 score, the ROC curve has also been utilized. The model was then compared to seven other well-known ML classifiers which includes Naive Bayes, SVM, AdaBoost, k-NN, Random Forest, Decision Tree, and Linear Regression, to produced its performance. According to the proposed model, it overcomes existing approaches by 97.65%, highlighting its high accuracy.

The state-of-the-art analysis reveals that the majority of studies on IDS rely on benchmark datasets such as KDD99 and NSL-KDD. Nevertheless, these datasets are considered outdated and do not encompass current attack scenarios. Additionally, only a limited number of papers create their own real-time datasets to assess the intrusion detection capabilities of their IDS models within a network. Furthermore, the evaluation of intrusion detection rates in most IDS studies primarily focuses on traditional machine learning (ML) models, particularly for identifying DoS or DDoS attacks. The primary objective of our proposed research is to mitigate the time taken for detecting attacks and achieve a high detection rate and classify attacks in real-time, with a focus on improving the detection rate as well.

3. Methodology

This section introduces the proposed Intrusion Detection System (IDS) model that aims to optimize the detection rate (DR) and FPR in a network. Figure 1. represents the working methodology of the proposed IDS model.

3.1. Network Flow Extraction

Initially, self generated normal network traffic data and malignant network traffic data are collected. Later, we proceeded to retrieve statistical features from both the normal and malignant PCAPs using the CICFlowMeter tool [23]. This tool has the ability to extract more than 80 network flow characteristics from the data. The Network traffic is devided into five attributes: Source IP, Destination IP, Source port, Destination port, and Protocol. The CICFlowMeter tool inspects these properties in the network flows and enumerates several statistical features. These features contribute insights into the characteristics and behavior of the network traffic, helping to classify normal and malignant activities. By leveraging the CICFlowMeter tool and to extract statistical network flow features, the we gathered important information that will aid in succeeding steps of the proposed IDS model, such as feature selection and classification algorithm implementation.

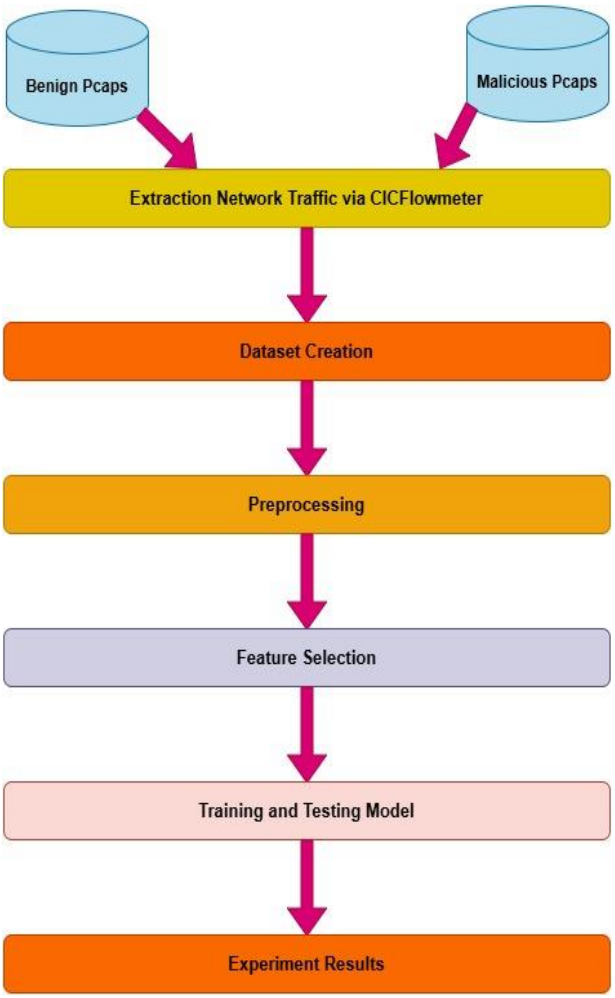

Figure 1. Flow diagram of the proposed methodology

3.2 Dataset Creation

After extracting the statistical network flow features, the dataset is labeled to differentiate between specific malware and benign network flow traffic. The labeling process involves categorizing the network flows into three classes: Benign, Denial-of-Service (DoS), and Distributed Denial-of-Service (DDoS).

3.3 Data Pre-processing

Data preprocessing is a paramount task for dropping and manipulating raw data before it is used to enrich the performance of classifiers. The primary motive of preprocessing is to mitigate the duplicate records and unrelated features of the dataset. Three steps were followed for preprocessing the dataset.

3.4 Data Cleaning

In this study, the self-created dataset was used for evaluating the IDS model. The dataset contained 2867 nan and infinity records, and these were removed from the dataset. We used 636729 instances of normal or benign attack and all instances of attacks to evaluate the IDS model. The statistics of attacks are listed in Table 1. The total number of records 1193285 were utilized for building the IDS model.

Table 1. Statistics of Benign and Malicious flow

Class/Attack Types	Number of Instances
Benign	6,36729
Malicious	5,56556
Total	11,93285

3.5 Label Encoding

One categorical label feature and several numerical features are included in this dataset. Label encoding denotes the process of transforming a label into a numerical representation the is machine interpretable. ML algorithms can then make superior decisions about how those labels must be manipulated. It is a significant preprocessing phase for structured datasets in supervised learning. The label value must be converted into a vector of numeric values. In the context of label feature for multi-class classification; benign, DoS, and DDoS are converted into 0, 1, and 2 respectively.

3.6 Normalization

Normalization in machine learning defines to the mechanism of transforming input data into a standardized range or distribution. It is a crucial preprocessing step that helps enhance the performance and convergence of many ML algorithms. Normalization ensures that all input features or variables have similar scales, preventing some features from dominating or biasing the learning process. There are several common methods for normalization in machine learning:

1. Min-Max Scaling (Normalization): This approach rescales the data to a fixed range, usually between 0 and 1. It can be represented as [24].

$$X\_scaled = (X - X\_min) / (X\_max - X\_min) \quad (1)$$

Where X is the original value, X_scaled is the scaled value, X_min is the minimum value in the dataset, and X_max is the maximum value in the dataset.

2. Z-Score (Standardization): This apporach involves standardizing the data to achieve a mean of 0 and a standard deviation of 1. The mathematical expression for standardizing using z-score can be represented as [25].

$$X\_scaled = (X - X\_mean) / X\_std \quad (2)$$

We have used z-score to normalize the dataset. Some attributes in our dataset, for example 'Flow duration', 'Fwd packet length std', 'Flow bytes / s', 'Flow packets / s', and 'Flow IAT mean', exhibit a significant difference between their maximum and minimum values. It indicates a substantial variation in the scales of these attributes. To address this issue, it is advisable to apply normalization techniques like Min-Max Scaling or Z-Score Standardization. These methods help bring the attributes to a comparable scale, ensuring that their discrepancies do not disproportionately affect ML algorithms. By normalizing the data, you can enhance the accuracy and reliability of your models when functioning with these attributes. To normalize the values of these attributes have been attained in the range of [0, 1].

3.7 Feature Selection

It is a crucial procedure in building predictive models, involving the selection of a subset of essential features or variables from a bigger set of available features. The purpose of feature chosen is to identify the most meaningful and impactful features that contribute to the predictive power of a model while eliminating irrelevant or redundant features. This is important because including too many features in a model can lead to overfitting, that signifies the model will implement well on the training data nonetheless unwell on new and unrecognized data. There are various methods for feature selection, some of them discussed.

3.7.1 Filter methods

These methods rely on statistical measures to rank the importance of features. The most commonly utilized filters are correlation-based feature selection, mutual information-based feature selection, and chi-squared feature selection.

3.7.2 Wrapper methods

These methods estimate the performance of the model with diverse subgroups of features. This is a more computationally expensive approach, as it involves training and evaluating the model multiple times.

3.7.3 Embedded methods

These methods select features as portion of the model-building process. For example, some ML algorithms, such as Lasso and Elastic Net, have built-in feature selection mechanisms.

Overall, feature selection is significant process in the model-building process because it can help improve the performance of a predictive model by mitigating the number of features used. This research utilized the RFE using a RF classifier for selecting the appropriate features. The RF classifier was modified to utilize better parameters to select the relative subset of features. The parameters are summarized in Table 2. Moreover, RFE algorithm measures the performance of the RF classifier for each subset of features in an iterative manner. According to the RF classifier's better performance, the RFE selects the best subset of features and omits the weakest subset. The iteration takes place until a set of features that is considered the most effective is achieved. The list of selected features is illustrated in Table 3 with types.

Table 2. Parameters used by RF in RFE

Classifiers	Parameters
RF	n_estimators: 45 random_state: 23 max_depth: 26 n_jobs: -1

Table 3. List of selected features

S.No	Features
1	Destination Port
2	Total Length of Fwd Packets
3	Bwd Packet Length Mean
4	Average Packet Size
5	Fwd Header Length
6	Packet Length Variance
7	Avg Bwd Segment Size
8	Fwd IAT Mean
9	Init_Win_bytes_forward
10	Init_Win_bytes_backward

4. Model Construction

Model construction is a crucial and essential phase while identifying any threats in network. Here, we provided two phases for constructing the proposed IDS model. In first phase, we discussed about the selection of classifier and used those classifier based on literature survey which will give fruitful results. In second phase, experimental set up and important simulation parameters are discussed while making the IDS model.

4.1 Selection of classifiers

Random Forest is a versatile ML algorithm utilized for tasks such as classification, regression, and more [26]. It belongs to the ensemble learning family, utilizing a combination of multiple decision trees to form a forest. The algorithm operates by making a set of decision trees using a randomly designated subclass of the training data and an arbitrarily selected subclass of the features for each tree. During the training phase, each tree is built utilizing a different subset of the training data, and every node of the tree is split utilizing the top feature and split point among a random subset of features.

Decision tree algorithm is a popular ML algorithm utilized for solving classification and regression problems [27]. It operates by recursively partitioning the data into subsets according to the most significant attribute or feature at each node of the tree. The resulting decision tree can be visualized as a tree structure where each internal node enacts an attribute, each branch also enacts a value for that attribute, and each leaf node denotes a particular label.

One of the advantages of DT is that they are simple to interpret and can be used to attain insights into the relationships between the attributes and the target variable. However, they are prone to overfitting and may not perform well on noisy or complex datasets. Techniques such as pruning, ensemble methods (such as random forests), and boosting can be utilized to overcome these limitations.

Extra Trees Classifier (ETC) is an ensemble learning method used for classification tasks [28]. It is similar to the Random Forest Classifier (RFC) in that it builds numerous decision trees and amalgamates their predictions to make the final decision [29-30]. However, unlike RFC, ETC randomly selects splitting points for each feature and uses the whole training set to grow each decision tree, rather than using a random subset of features and samples.

K-Nearest Neighbors (KNN) is also ML based algorithm applied for both classification and regression problems [31-35]. It is a kind of instance-based learning, where the algorithm does not learn a model but instead memorizes the training instances and makes prophecies according to the similarity between new instances and the training instances[36-39].

4.2 Training and Testing

We use Python programming language and Colab Jupyter notebook for constructing the efficient and robust IDS model. The experiment was conducted on a HP series laptop with a 64-bit Intel-i5 processor running at 2.3 GHz, and 16GB of RAM. Approximately 80% of the data in the total dataset has been utilized for training, while 20% is used for testing. We utilized four ML classifiers RF, DT, ET, and K-NN to construct the IDS model. As shown in Table 4, the simulation parameters are summarized in all used classifiers. All classifiers are employed for building an IDS model.

Table 4. Parameters used in constructing model

S.No	Features
RF	n_estimators: 23, random_state: 23,max_depth: 23, n_jobs: -1
DT	random_state: 3
ET	n_estimators: 25, random_state: 23, n_jobs: -1
KNN	n_nieghbors: 3

4.3 Performance Evaluation Criteria

The proposed IDS model is evaluated based on different performance metrics. A detailed analysis of the selected performance metrics from (3) to (7).

$$\text{Accuracy} = \frac{TP+TN}{TP+FP+TN+FN} \quad (3)$$

$$\text{Recall} = \frac{TP}{TP+FN} \quad (4)$$

$$\text{Precision} = \frac{TP}{TP+FP} \quad (5)$$

$$\text{FPR} = \frac{FP}{Fp+TN} \quad (6)$$

5. Experiment Results

After implementing the several classifiers in the self-created dataset, the results were explored in terms of different statistical measures as illustrated in Table 5. It can be observed that the accuracy of the all classifier are high for the proposed model. However, the least accurate is k-NN. FPR and detection time are also low which is very

significant for achieving the malware detection in real time. Moreover, confusion matrices are also represented in Figure 2, Figure 3, Figure 4, and Figure 5 respectively.

A confusion matrix is a technique for describing the performance of a statistical classification model. Classification accuracy itself can be misleading if the number of observations in each class is not the same, or if the dataset has more than two classes. Calculating a confusion matrix can gives a better understanding of the classification model's correct behaviour. Figure 6 presents a comparison between the two key aspects: model building time (sec) and detection time (sec). It is observed that the proposed ML based model perform better than the existing studies [40-41] the detection time of the IDS model has been calculated using the formula as shown in equation 7.

$$\text{Detection Time (DT)} = \text{End Time(ET)} - \text{Start Time(ST)} \quad \text{.....(7)}$$

Table 5. Comparison of different ML algorithms using self-created dataset

Model	Accuracy (%)	Recall (%)	Precision (%)	F1-score (%)	FPR (%)	Building Time(sec)	Detection Time (sec)
DT	99.96	99.96	99.98	99.97	0.0002	3.5	0.001
ET	99.96	99.97	99.98	99.97	0.0002	3.6	0.10
k-NN	98.82	98.76	99.48	99.11	0.01	30.87	0.003
RF	99.97	99.97	99.98	99.97	0.0001	6.10	0.10
Monika et.al [40]	97.1	99.1	96	97.5	-	-	-
Neeraj et.al [41]	98.92	98.35	99.47	98.80	0.52	-	-

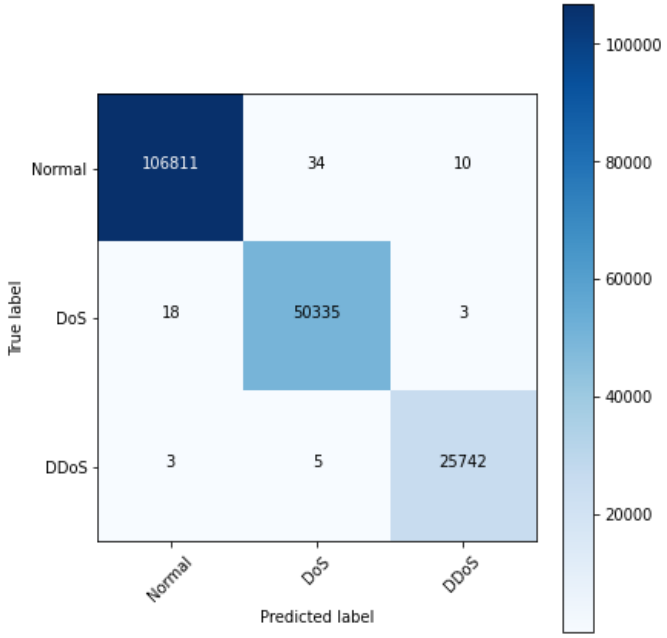

Figure 2. Confusion Matrix of DT

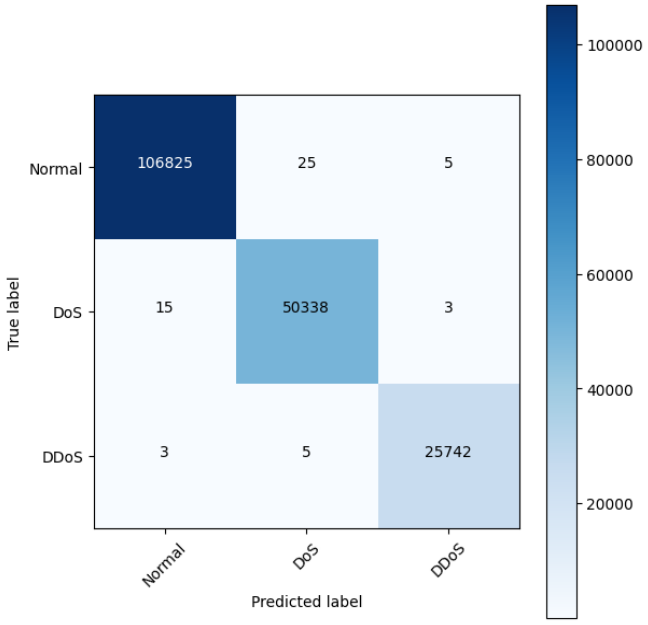

Figure 3. Confusion Matrix of ET

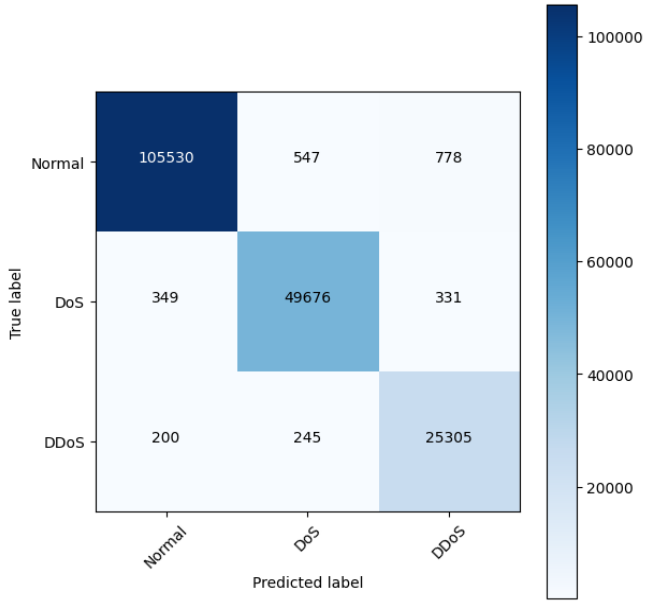

Figure 4. Confusion Matrix of k-NN

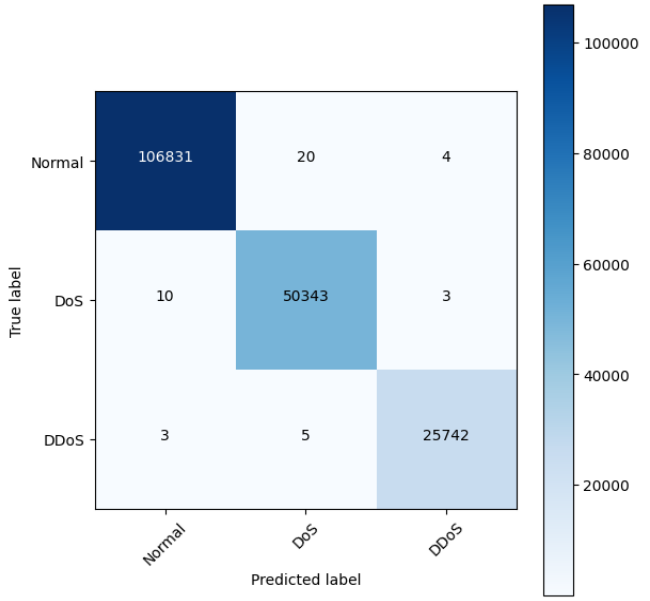

Figure 5. Confusion Matrix of RF

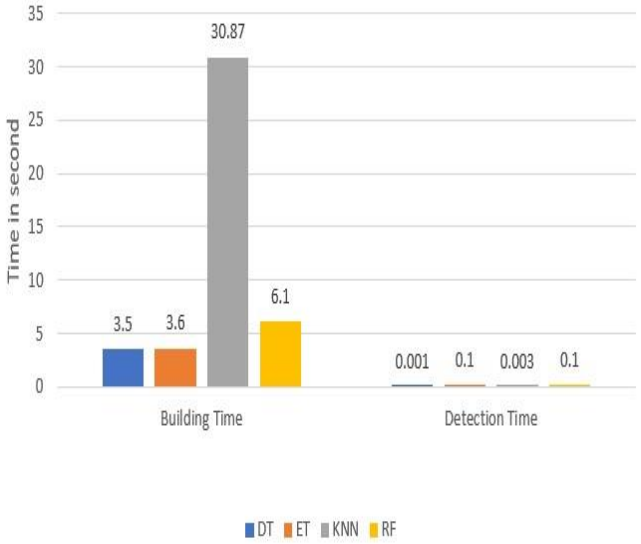

Figure 6. Building Time and Detection Time Comparison

6. Conclusion

Edge computing plays a significant role in performing real-time data computation and preprocessing. However, this system is susceptible to various attacks employed by intruders or attackers to disrupt the services provided by edge computing. This paper highlights the significance of securing and maintaining the integrity of data in edge computing systems. In addition, we proposed an IDS model for edge computing system which assesses its effectiveness using a self-created dataset that encompasses common attacks such as DoS, DDoS, as well as normal network behavior. The evaluation of the proposed IDS model reveals superior performance when compared to existing traditional ML based IDS models. Several evaluation metrics, including Recall, Precision, DR and FPR demonstrate higher levels of performance. It can be concluded that the proposed IDS based model operates effectively as "vulture eyes" within an edge computing network, enabling the immediate detection of malicious attacks. The advacnce deep learning algorithms will be utilized in future work for better scalability of the model.

Author Contributions:
Abhay Pratap Singh Bhadauria: Conceptualization, Methodology, Software **Mahendra Singh:** Data curation, Writing-Original draft preparation, Software, Validation. **Rajesh Kumar:** Visualization, Investigation, Writing-Reviewing and Editing. **Amit Kumar:** Writing-Reviewing and Editing.

Conflicts of interest
The authors declare no conflicts of interest.

References

1. King, J., & Awad, A. I. (2016). A distributed security mechanism for resource-constrained IoT devices. Informatica, 40(1), 133-143.
2. Weber, M., & Boban, M. (2016). Security challenges of the internet of things. In 2016 39th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO), 638–643. https://10.1109/MIPRO.2016.7522219
3. Dirik, M. (2023). Machine learning-based lung cancer diagnosis. Turkish Journal of Engineering, 7(4), 322–330. https://doi.org/10.31127/tuje.1180931
4. Cao, K., Liu, Y., Meng, G., & Sun, Q. (2020). An overview on edge computing research. IEEE Access, 8, 85714–85728. https:// 10.1109/ACCESS.2020.2991734
5. Liu, B., Luo, Z., Chen, H., & Li, C. (2022). A survey of state-of-the-art on edge computing: Theoretical models, technologies, directions, and development paths. IEEE Access, 10, 54038–54063. https://10.1109/ACCESS.2022.3176106
6. Bakhsh, S. A., Khan, M. A., Ahmed, F., Alshehri, M. S., Ali, H., & Ahmad, J. (2023). Enhancing IoT network security through deep learning-powered intrusion detection system. Internet of Things, 24, 100936. https://doi.org/10.1016/j.iot.2023.100936

7. Ghazal, T. M., Hasan, M. K., Alzoubi, H. M., Alshurideh, M., Ahmad, M., & Akbar, S. S. (2023). Internet of Things connected wireless sensor networks for smart cities. In The effect of information technology on business and marketing intelligence systems, 1056, 1953–1968, Cham: Springer International Publishing. https://doi.org/10.1007/978-3-031-12382-5_107
8. Neto, E. C. P., Dadkhah, S., Ferreira, R., Zohourian, A., Lu, R., & Ghorbani, A. A. (2023). CICIoT2023: A real-time dataset and benchmark for large-scale attacks in IoT environment. Sensors, 23(13), 5941. https://doi.org/10.3390/s23135941
9. Falayi, A., Wang, Q., Liao, W., & Yu, W. (2023). Survey of distributed and decentralized IoT securities: Approaches using deep learning and blockchain technology. Future Internet, 15(5), 178. https://doi.org/10.3390/fi15050178
10. Shao, S., Zheng, J., Guo, S., Qi, F., & Qiu, I. X. (2023). Decentralized AI-enabled trusted wireless network: A new collaborative computing paradigm for the Internet of Things. IEEE Network, 37(2), 54–61. https:// 10.1109/MNET.002.2200391
11. Song, F., Ma, Y., Yuan, Z., You, I., Pau, G., & Zhang, H. (2023). Exploring reliable decentralized networks with smart collaborative theory. IEEE Communications Magazine, 61(8), 44–50. https:// 10.1109/MCOM.003.2200443
12. Li, T., Yu, L., Ma, Y., et al. (2023). Carbon emissions of 5G mobile networks in China. Nature Sustainability, 6(12), 1620–1631. https://doi.org/10.1038/s41893-023-01206-5
13. Kao, H. W., & Wu, E. H. K. (2023). QoE sustainability on 5G and beyond 5G networks. IEEE Wireless Communications, 30(1), 118–125. https:// 10.1109/MWC.007.2200260
14. Gendreau, A. A., & Moorman, M. (2016). Survey of intrusion detection systems towards an end-to-end secure Internet of Things. In 2016 IEEE 4th International Conference on Future Internet of Things and Cloud (FiCloud), 84–90. https:// 10.1109/FiCloud.2016.20
15. Packet Total - A useful site for analyzing PCAP files. (n.d.). Bleeping Computer. Retrieved December 14, 2023, from https://www.bleepingcomputer.com/news/security/packettotal-a-useful-site-for-analyzing-pcap-files/
16. Wireshark. (n.d.). Retrieved December 7, 2023, from https://www.wireshark.org/
17. Banerjee, U., Vashishtha, A., & Saxena, M. (2010). Evaluation of the capabilities of Wireshark as a tool for intrusion detection. International Journal of Computer Applications, 6(7), 1–5.
18. Canadian Institute for Cybersecurity (CIC). (n.d.). Retrieved December 7, 2024, from https://www.unb.ca/cic/datasets/index.html
19. Loganathan, G., Samarabandu, J., & Wang, X. (2018). Real-time intrusion detection in network traffic using adaptive and auto-scaling stream processor. In 2018 IEEE Global Communications Conference (GLOBECOM), 1–6. https:// 10.1109/GLOCOM.2018.8647489
20. Rathore, M. M., Paul, A., Ahmad, A., Rho, S., Imran, M., & Guizani, M. (2016). Hadoop-based real-time intrusion detection for high-speed networks. In 2016 IEEE Global Communications Conference (GLOBECOM), 1–6. https:// 0.1109/GLOCOM.2016.7841864
21. Lee, S. J., Yoo, P. D., Asyhari, A. T., Jhi, Y., Chermak, L., Yeun, C. Y., & Taha, K. (2020). IMPACT: Impersonation attack detection via edge computing using deep autoencoder and feature abstraction. IEEE Access, 8, 65520–65529. https:// 10.1109/ACCESS.2020.2985089
22. Shaikh, A., & Gupta, P. (2022). Real-time intrusion detection based on residual learning through ResNet algorithm. International Journal of System Assurance Engineering and Management, 1–15. https://doi.org/10.1007/s13198-021-01558-1
23. Singh, A. P., Singh, M., Bhatia, K., Pathak, H. (2024). Encrypted malware detection methodology without decryption using deep learning-based approaches. Turkish Journal of Engineering, 8(3), 498-509. https://doi.org/10.31127/tuje.1416933
24. Raju, V. G., Lakshmi, K. P., Jain, V. M., Kalidindi, A., & Padma, V. (2020). Study the influence of normalization/transformation process on the accuracy of supervised classification. In 2020 Third International Conference on Smart Systems and Inventive Technology (ICSSIT), 729–735. https:// 10.1109/ICSSIT48917.2020.9214160
25. Patro, S. G. O. P. A. L., & Sahu, K. K. (2015). Normalization: A preprocessing stage. arXiv. https://arxiv.org/abs/1503.06462
26. Pradhan, D., & Muduli, D. (2023). Software defect prediction model using AdaBoost-based random forest technique. In 2023 14th International Conference on Computing Communication and Networking Technologies (ICCCNT), 1–6. https:// 10.1109/ICCCNT56998.2023.10308208
27. Maurya, A., & Gaur, S. (2023). A decision tree classifier-based ensemble approach to credit score classification. In 2023 International Conference on Computing, Communication, and Intelligent Systems (ICCCIS), 620–624. https:// 10.1109/ICCCIS60361.2023.10425039
28. Khandelwal, N., & Sakalle, V. (2024). A review of customer churn prediction in telecommunications and the medical industry using machine learning classification models. International Journal of Innovative Research in Technology and Science, 12(2), 366–379.
29. Basholli, F.,Mema, B.,& Basholli, A. (2024). Training of information technology personnel through simulations for protection against cyber-attacks. Engineering Applications, 3(1), 45-58
30. Leka, B., & Hoxha, K. (2024). Software engineering methodologies in programming companies in Albania. Engineering Applications, 3(1), 85-91
31. Pradhan, D., & Muduli, D. (2023). Software defect prediction model using AdaBoost-based random forest technique. In 2023 14th International Conference on Computing Communication and Networking Technologies (ICCCNT), 1–6. https:// 10.1109/ICCCNT56998.2023.10308208
32. Mogaraju, J. K. (2024). Machine learning empowered prediction of geolocation using groundwater quality variables over YSR district of India. Turkish Journal of

Engineering, 8(1), 31-45. https://doi.org/10.31127/tuje.1223779
33. Raman, R., Kantari, H., Gokhale, A. A., Elangovan, K., Meenakshi, B., & Srinivasan, S. (2024). Agriculture yield estimation using machine learning algorithms. In 2024 International Conference on Automation and Computation (AUTOCOM), 187–191. https://10.1109/AUTOCOM60220.2024.10486107
34. Juraev, D. A., Elsayed, E. E., Bulnes, J. J. D., Agarwal, P., & Saeed, R. K. (2023). History of ill-posed problems and their application to solve various mathematical problems. Engineering Applications, 2(3), 279–290. https://publish.mersin.edu.tr/index.php/enap/article/view/1178
35. Mema, B., & Basholli, F. (2023). Internet of Things in the development of future businesses in Albania. Advanced Engineering Science, 3, 196–205. https://publish.mersin.edu.tr/index.php/ades/article/view/1325
36. Demiröz, A., Barstugan, M. ., Saran, O., & Battal, H. (2023). Determination of compaction parameters by image analysis technique. Advanced Engineering Science, 3, 137–150. https://publish.mersin.edu.tr/index.php/ades/article/view/1192
37. Kocalar, A. C. (2023). Sinkholes caused by agricultural excess water using and administrative traces of the process. Advanced Engineering Science, 3, 15-20
38. Naumov, A., Khmarskiy, P., Byshnev, N., & Piatrouski, M. (2023). Methods and software for estimation of total electron content in ionosphere using GNSS observations. Engineering Applications, 2(3), 243–253. Retrieved September 14, 2024, from https://publish.mersin.edu.tr/index.php/enap/article/view/1165
39. Meghraoui, K., Sebari, I., Bensiali, S., & Ait El Kadi, K. (2022). On behalf of an intelligent approach based on 3D CNN and multimodal remote sensing data for precise crop yield estimation: Case study of wheat in Morocco. Advanced Engineering Science, 2, 118–126. Retrieved September 14, 2024, from https://publish.mersin.edu.tr/index.php/ades/article/view/329
40. Vishwakarma, M., & Kesswani, N. (2023). A new two-phase intrusion detection system with Naïve Bayes machine learning for data classification and elliptic envelope method for anomaly detection. Decision Analytics Journal, 7, 100233. https://doi.org/10.1016/j.dajour.2023.100233
41. Saini, N., Bhat Kasaragod, V., Prakasha, K., & Das, A. K. (2023). A hybrid ensemble machine learning model for detecting APT attacks based on network behavior anomaly detection. Concurrency and Computation: Practice and Experience, 35(28), e7865. https://doi.org/10.1002/cpe.7865