ML-Driven Lightweight Botnet Detection System for IoT-Networks

Ashfaq Hussain Farooqi[*1], Rizwan Ahmad[1], Shaharyar Kamal[2]
[1]Department of Computer Science, Air University Islamabad, Pakistan.
[2]Department of Electrical Engineering, University of Chile, Santiago, Chile.
[*]**Correspondence:** ashfaq.hussain@au.edu.pk

Citation | Farooqi. A. H, Ahmad. R, Kamal. S, "ML-Driven Lightweight Botnet Detection System for IoT-Networks", IJIST, Special Issue. pp 194-206, Oct 2024
Received| Oct 11, 2024 **Revised|** Oct 16, 2024 **Accepted|** Oct 21, 2024 **Published|** Oct 26, 2024.

The integration of cloud computing with the Internet of Things (IoT) seeks to create seamless connections between humans and devices, enhancing applications in areas like smart healthcare and home automation. However, this also brings significant security challenges. Our study addresses the critical need for an efficient anomaly detection system specifically designed for IoT-enabled cloud computing environments, a gap not previously explored at this scale. Utilizing the IoT-23 dataset, we evaluated various feature selection techniques in conjunction with classification algorithms to develop a lightweight anomaly detection model. Our results demonstrate that the decision tree classifier, paired with the correlation coefficient method for feature selection, achieved an impressive 99.98% accuracy rate, with an average processing time of just 5.2 seconds. This combination proved to be the most effective for real-time anomaly detection, presenting a promising approach for ensuring robust security in IoT networks as connectivity continues to grow.

Keywords: Internet of Things (IoT), Intrusion Detection System (IDS), Machine Learning (ML), Feature Selection Algorithm, Botnet Detection.

Introduction:

Computing and mobile devices have become an integral part of daily life, with increasing dependence on this technology. As we look to the future, technological advancements toward the Metaverse are becoming increasingly evident, with a growing desire to adopt these emerging technologies. It is clear that future generations will live in a Multiverse [1], where today's mobile devices will evolve into a more immersive and interconnected experience. Future technologies will integrate a multitude of sensors on the human body and in the surrounding environment to augment a virtual experience within the physical world [2]. Moreover, the future will be defined by the ability to connect everything, allowing humans to control their environments through gestures and to live in multiple virtual and augmented worlds simultaneously. The ongoing advancements in 5G, IoT, cloud computing, edge computing, high-performance computing, blockchain, and AI hold immense potential to bring the Metaverse to life by connecting various IoT devices. However, a significant challenge for current IoT-enabled cloud computing environments is addressing security concerns [3], as the rise of these new technologies will dramatically expand the attack surface. With the Metaverse's emergence, not only will the attack surface increase, but the number of interconnected devices will grow exponentially, adding further complexity. Many of these devices are IoT-based, which presents additional difficulties in authenticating and authorizing connections [4]. Therefore, 6G and beyond must not only focus on network speed but also create a robust network that can support the Metaverse through the convergence of these technologies. This includes leveraging blockchain for zero-trust authentication and authorization [5][6], as well as incorporating AI at every level of communication. AI will play a critical role in the Metaverse, working in tandem with blockchain to facilitate a zero-touch environment. This would enable computer vision to interpret human gestures, emotions, and commands, machine learning (ML) for routing and end-to-end service optimization, AI-driven anomaly detection and mitigation [7], and large language models for automated configuration management. As such, AI and ML will be essential for addressing security threats in this future technological landscape.

Smart homes, as part of the larger IoT-enabled cloud computing ecosystem, are an important component of the Metaverse and its future applications. These systems enable service providers to monitor and control devices such as appliances, doors, windows, and other smart equipment, using sensors and processing capabilities to enhance user experience. However, with the proliferation of IoT devices, the risk of botnet attacks is growing [8]. Adversaries now have access to a variety of tactics and methodologies to launch different types of network attacks. This study focuses on evaluating different strategies to mitigate botnet attacks within IoT networks, exploring various feature extraction techniques to reduce the complexity of features, and applying AI-driven classification methods to distinguish between attack types and normal behavior. The IoT-23 dataset, which simulates a smart home environment with devices like door lockers, smart LEDs, and Echo IoT devices [9], is used to conduct a series of tests aimed at finding the most energy-efficient solution with the highest detection accuracy. This dataset is widely used by researchers to evaluate proposed intrusion detection system (IDS) solutions. Several IDS-based approaches have been proposed to mitigate internal attacks [10][11][12][13]. In our study, we assess feature selection techniques such as manual selection, the chi-square test, information gain, correlation coefficient, and random forest, alongside classification algorithms like support vector machines (SVM), Naive Bayes, and decision trees. It is crucial to examine how each algorithm identifies various attack forms and anomalies to understand their respective strengths and limitations. The evaluation is conducted using established performance metrics, including accuracy, precision, recall, F1-score, and processing time. Our findings show that the combination of the correlation coefficient method and the decision tree classifier offers the best performance in terms of accuracy and efficiency. We recommend positioning the IDS agent at the fog layer of the IoT-enabled cloud computing environment, as depicted in Figure 1. By

analyzing the traffic generated by IoT devices in smart homes at the fog layer, a distributed detection system can be established, enabling faster anomaly detection through localized data processing and analysis.

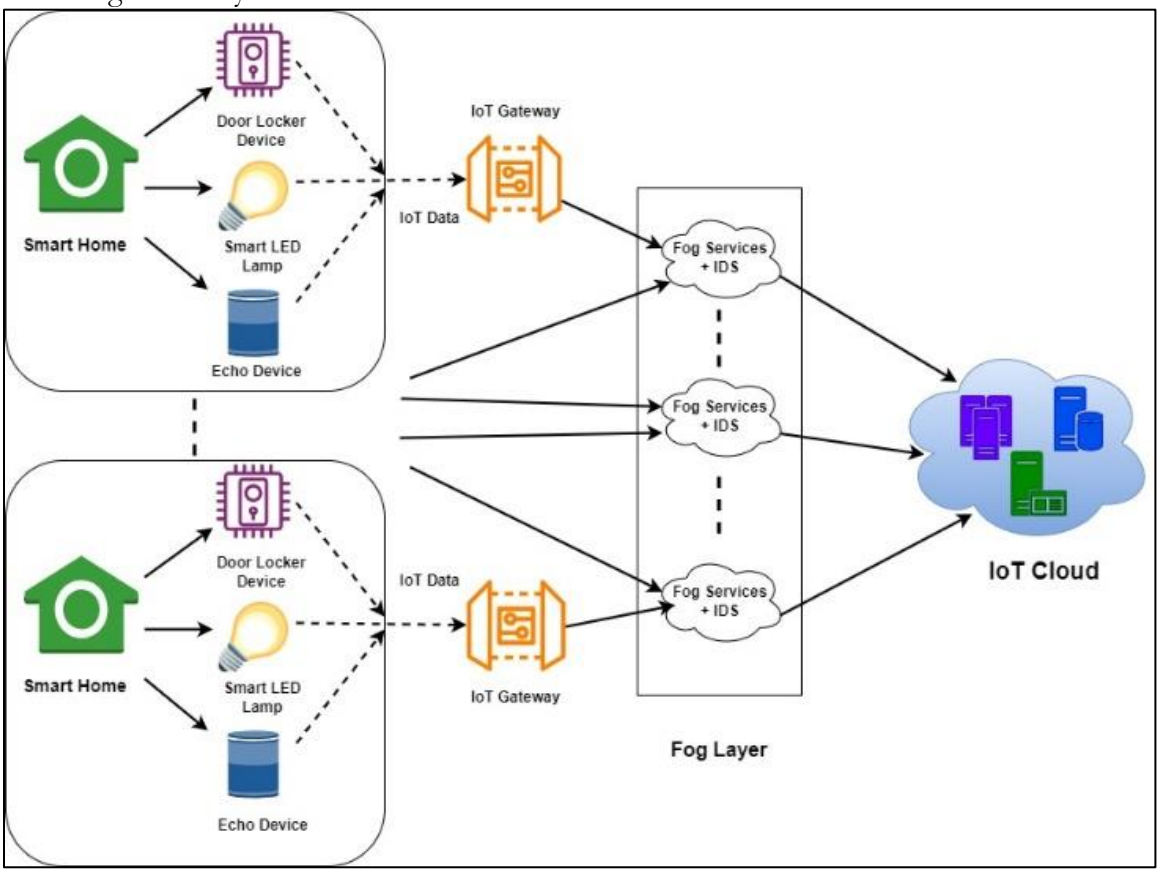

Figure 1. Architecture diagram illustrating the implementation of an intrusion detection system in an IoT Smart Home system

Novelty:

The techniques selected in this study are well-known machine learning (ML) methods, but the novelty lies in how they are combined to detect botnets in Internet of Things (IoT) networks. These specific combinations have not been explored in previous research. While using established performance metrics such as Accuracy, Precision, Recall, and F1-Score is standard practice, the innovation in our approach lies in how these metrics are applied to the newly generated IoT-23 dataset, employing various combinations of feature selection techniques and ML classifiers. Furthermore, unlike most studies that focus primarily on detection accuracy, our research emphasizes the importance of time utilization as a critical metric. This is particularly relevant for IoT applications, where computational resources and response times are often constrained. Our study provides a comprehensive evaluation that balances accuracy and computational efficiency, offering valuable insights for securing real-world IoT deployments where such trade-offs are essential.

Although the decision tree (DT) algorithm has been previously used for classification on the IoT-23 dataset, it achieved only around 70% accuracy. In contrast, our proposed approach improves the accuracy to over 98% by applying a specialized preprocessing technique followed by feature selection. Notably, the decision tree achieved the highest accuracy when using feature selection methods such as chi-square, random forest, or the correlation coefficient. However, the application of the correlation coefficient feature selection method resulted in faster processing times compared to chi-square, random forest, and information gain, making it a more efficient choice for real-time anomaly detection.

Research Objectives:

The goal is to develop a lightweight, machine learning (ML)-based anomaly detection system for IoT-enabled cloud computing environments that achieves high accuracy while minimizing time consumption.

Research Contributions:

This research makes a significant contribution by thoroughly assessing botnet attacks targeting various IoT devices in smart home environments, along with the evolving cybersecurity risks associated with IoT systems. The analysis is conducted using the IoT-23 dataset, which includes both botnet attack instances and benign samples. The key contributions of this study are as follows:

- A comprehensive comparison and evaluation of well-known feature selection techniques, including the Chi-Square test, information gain, correlation coefficient, and random forest.
- Experiments conducted on the state-of-the-art IoT-23 dataset, which is specifically designed to represent IoT network behaviors, including both attack and normal traffic.
- The use of widely recognized machine learning (ML) classification algorithms, such as SVM, Naive Bayes, and Decision Tree, following the application of feature selection methods, to identify the most effective approach for botnet detection.
- An evaluation of all combinations of feature selection and classification algorithms against established performance metrics, including Accuracy, Precision, Recall, F1-Score, and Time.

This paper is structured as follows: Section 2 presents the related work in the field, while Section 3 details the proposed methodology and explains its implementation. Section 4 covers the performance evaluation system, including the metrics used and the experimental results. Finally, Section 5 concludes the paper, summarizing the findings and suggesting potential future research directions.

Related Work:

Machine learning (ML) models can be trained to detect patterns of botnet activity within IoT-enabled cloud computing environments. These models analyze data such as network traffic, transactions, and interactions among devices to identify potential botnet behavior. Several ML-based anomaly detection systems have been proposed for various networking paradigms, including wireless sensor networks, software-defined networks, peer-to-peer networks, cyber-physical systems, and IoT [14]. In addition, deep learning (DL) classification techniques have been explored for intrusion detection systems (IDS) in these environments [15]. However, intrusion detection in IoT networks remains challenging due to data heterogeneity, device constraints, and the wide range of potential applications [16]. The ability of artificial intelligence, particularly machine learning, to handle data diversity and velocity has proven to be a powerful tool for addressing these IoT security challenges [17].

For anomaly-based intrusion detection in IoT backbone networks, Pajouh et al. [18] introduced a model with two levels of classification and two layers of dimensionality reduction. They applied Principal Component Analysis (PCA) and Linear Discriminant Analysis (LDA) for feature reduction, followed by a certainty-factor k-nearest neighbor classifier as a secondary check after the Naïve Bayes classifier categorized the traffic. This system achieved 84.86% accuracy, with a false positive rate (FPR) of 4.86%, although the detection time was not provided. In [19], the authors focused on Mirai botnet detection, one of the most prevalent IoT botnet attacks. They developed a mining-based detection method leveraging deep learning and neural network models to identify variations of the Mirai botnet, such as Hakai. Using the IoT-23 dataset, their model achieved around 90% accuracy. A one-class KNN classifier-based approach for detecting IoT botnets in heterogeneous environments was proposed in [20],

demonstrating accurate and rapid detection of IoT botnets in their early stages. A combination of deep learning and three-level algorithms for fast and accurate attack detection in IoT networks was introduced in [21]. Evaluated using the IoT-23 dataset, the method showed significant improvements in detection performance over previous approaches.

In another study, [22] examined the use of computer vision to detect and categorize attacks in the IoT-23 dataset. The researchers found that computer vision could perform effectively and efficiently on a Jetson Nano platform. For software-defined IoT networks, Li et al. [23] proposed an IDS based on the Bat algorithm and artificial intelligence. This flow-based IDS recorded network flows to detect attacks, achieving 96.42% detection accuracy with an FPR of 0.98%. However, the system's time overhead increased significantly with the number of flows, reaching nearly 4.5 seconds for 5104 flows. In [24], the authors introduced a Dense Random Neural Network-based intrusion detection framework for IoT networks, achieving detection rates of 99.14% for binary classification and 99.05% for multi-class classification. Kamaldeep et al. [25] developed IoT-Sentry, an IDS for IoT's cross-layer, which demonstrated a 99.46% accuracy rate when tested on a non-standard dataset. However, IoT-Sentry is limited in scope, defending against only a small number of threats and relying solely on packet-based characteristics, with no detailed description of time overhead.

Jeelani et al. [26] proposed an IoT anomaly detection system using the IoT-23 dataset, utilizing different ML and DL algorithms. Their model achieved 69% accuracy using SVM. This result motivated our research to experiment with various ML models in combination with feature selection techniques to improve detection accuracy and efficiency, aiming to provide a more effective solution for botnet detection in IoT environments

Material and Method:

This study proposes a methodology for implementing a machine learning (ML)-driven, lightweight botnet detection system, which is evaluated using the IoT-23 dataset. The proposed approach is visually outlined in **Figure 2**, which illustrates the key components and workflow of the methodology.

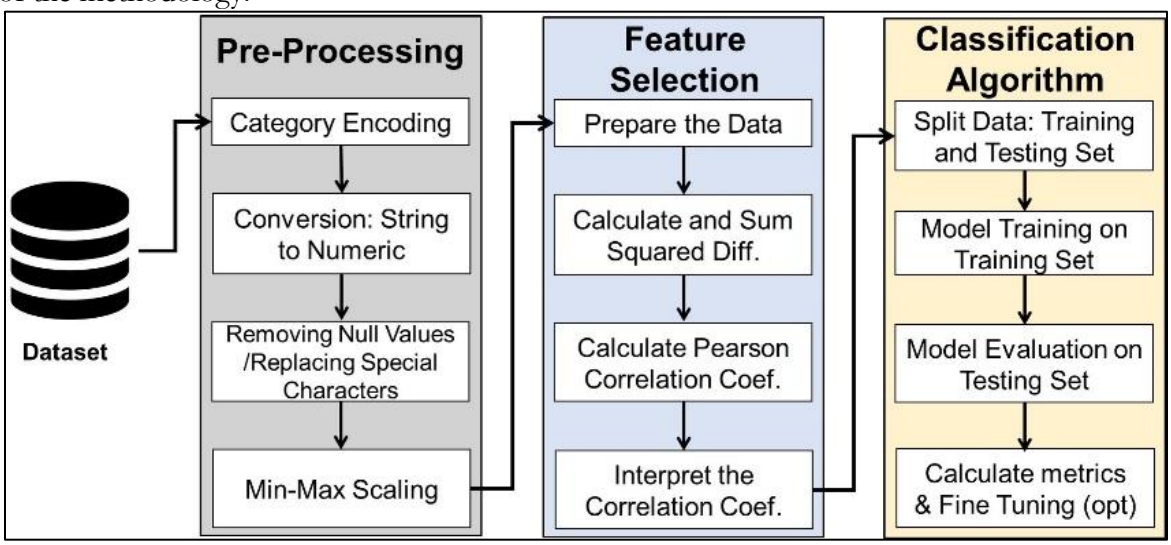

Figure 2. Proposed methodology for detecting botnet attacks in IoT environment.

Firstly, pre-processing is applied to prepare the data for feature extraction. Secondly, feature selection method is applied to find the features that can be only observed to classify between multiple classes in the classification phase. The detail of each step is discussed below:

Pre-Processing:

Preprocessing is the first crucial step before applying a classification algorithm, as raw data typically leads to poor performance. Most algorithms cannot handle string data or missing (null) values, which can severely impact the results. In our proposed methodology, we

implemented several preprocessing steps to prepare the dataset. First, category encoders are applied to convert string values into numeric values. Features with an "object" data type, containing string data, are transformed into either float or integer data types, as shown in **Table 1**. Since our dataset contains millions of records, it is impractical to manually assign a numeric value to each entry. For example, the "uid" feature contains millions of unique IDs, making manual encoding unfeasible [27][28]. To address this, we used category encoders, a collection of scikit-learn-style transformers designed to convert categorical variables into numeric data using different encoding methods. Specifically, we applied the **count encoding** method, which transforms categorical variables by assigning them a numeric value based on their frequency in the dataset. Common categories receive higher values, while rarer categories are assigned lower values.

Next, null values are removed, and records containing special characters are converted to zero. Finally, Min-Max Scaling is applied to rescale the data to a specific range, typically between 0 and 1, ensuring that all variables are on a comparable scale. This step standardizes the data and prepares it for effective processing by classification algorithms.

Table 1. Types of data before and after pre-processing in the proposed methodology

Data Item	Type (Before)	Type (Before)
Ts	int64	int64
uid, history, tunnel_parents	object	int64
id.orig_h, id.resp_h, proto, service, duration, orig_bytes, resp_bytes, conn_state, local_orig, local_resp, label, detailed-label	object	float64
id.orig_p, id.resp_p, missed_bytes, orig_ip_bytes	float64	float64
orig_pkts, resp_pkts, resp_ip_bytes	float64	int64

The sample size used for the analysis consists of 2,449,450 records from the IoT-23 dataset, representing 12% of the total dataset. This subset includes multiple classes of benign and anomalous behavior, such as benign, DDoS, C&C, attack, C&C-heartbeat, okiru, okiru-attack, and port scan, as detailed in **Table 2**. After preprocessing, the dataset was split into two parts: 80% of the data was used for training, while the remaining 20% was reserved for testing the model's performance.

Table 2. Distribution of attack types in the selected records for analysis

Attack Types	Port Scan	Okiru	Benign	DDoS	C&C	Attack	C&C-Heartbeat
Count	1377769	500707	286696	264215	15040	4493	481

Feature Selection (FS):

Feature selection is a machine learning technique used to enhance accuracy by improving the predictive power of algorithms. It achieves this by focusing on the most relevant features and removing unnecessary or irrelevant ones, thereby optimizing the model's performance. This highlights the importance of feature selection in the overall process. In the second phase of our proposed methodology, feature selection is applied to retain only the most meaningful features, ensuring that the subsequent classification phase is based on the most informative data.

Manual Selection:

Before performing classification, it is essential to select the most important features, as this choice significantly influences the classification results. In our proposed methodology, we begin by manually selecting features based on our domain knowledge to ensure the most relevant and impactful attributes are used. The selected features are listed in **Table 3**.

Chi Square Test:

Multiple machine learning algorithms are applied for feature selection, with the first being the Chi-Square method. The Chi-Square test is a statistical technique used to determine

whether two events are independent. By comparing the observed count (O) and the expected count (E) for two variables, the Chi-Square test calculates the difference between these values. When selecting features, our objective is to identify those that are most strongly related to the target outcome. If two features are independent, the observed count will closely match the expected count, resulting in a smaller Chi-Square value. Conversely, a higher Chi-Square value suggests that the independence hypothesis is incorrect, indicating a strong relationship between the feature and the outcome. Therefore, features with higher Chi-Square values—reflecting greater dependence on the response—are selected for model training. The Chi-Square test identifies 12 features, which are listed in **Table 3**.

Table 3. Number of selected features by the FS Technique from IoT-23 Dataset

FS Technique	Features	Features Selected
Manual	12	proto, service, duration, orig_bytes, resp_bytes, conn_state, missed_bytes, orig_pkts, orig_ip_bytes, resp_pkts, resp_ip_bytes, detailed-label
Chi-Square	12	ts, service, id.resp_p, orig_bytes, resp_bytes, conn_state, missed_bytes, id.orig_h, orig_ip_bytes, resp_pkts, id.resp_h, history
Information Gain	13	id.orig_h, id.orig_p, id.resp_h, id.resp_p, Proto, Service, conn_state, History, orig_pkts, orig_ip_bytes, Duration, ts, detailed-label
Correlation Coefficient	13	id.orig_h, id.orig_p, id.resp_h, id.resp_p, proto, service, duration, orig_bytes, resp_bytes, conn_state, missed_bytes, ts, detailed-label
Random Forest	13	ts, id.orig_p, id.resp_p, duration, orig_pkts, orig_ip_bytes, resp_ip_bytes, id.orig_h, id.resp_h, conn_state, history, label, detailed_label

Information Gain:

Information gain was the third feature selection method we employed. This approach evaluates the amount of information each variable contributes in relation to the target variable, making it useful for feature selection. It calculates the difference in entropy before and after the split, highlighting the imbalance in class distributions. The information gain values for each feature are shown in **Figure 3**. Based on these calculations, the features selected using information gain are listed in Table 3.

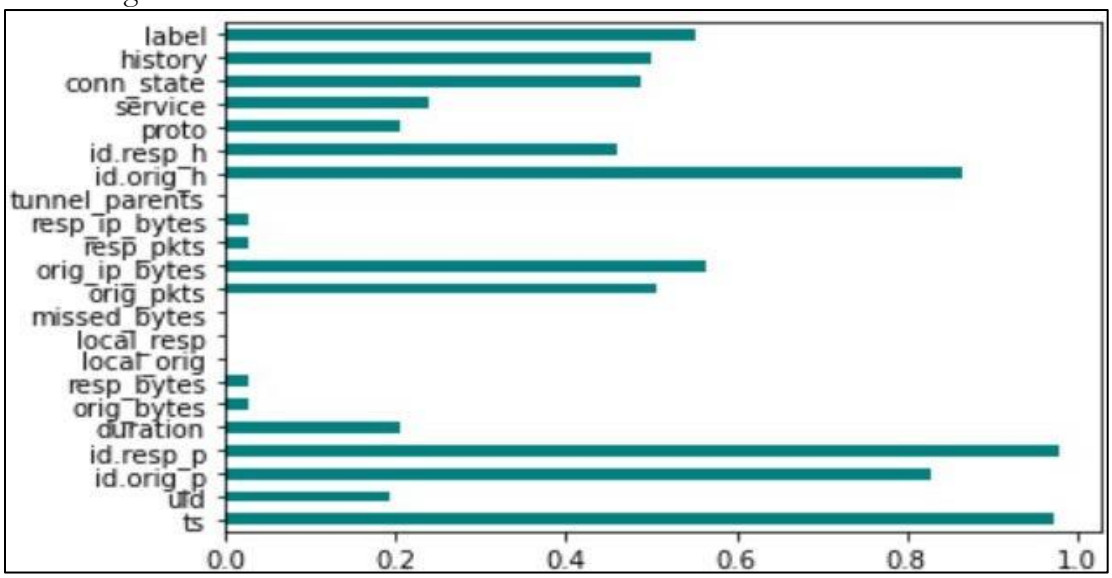

Figure 3. Information gain values for each feature obtained during the FS process

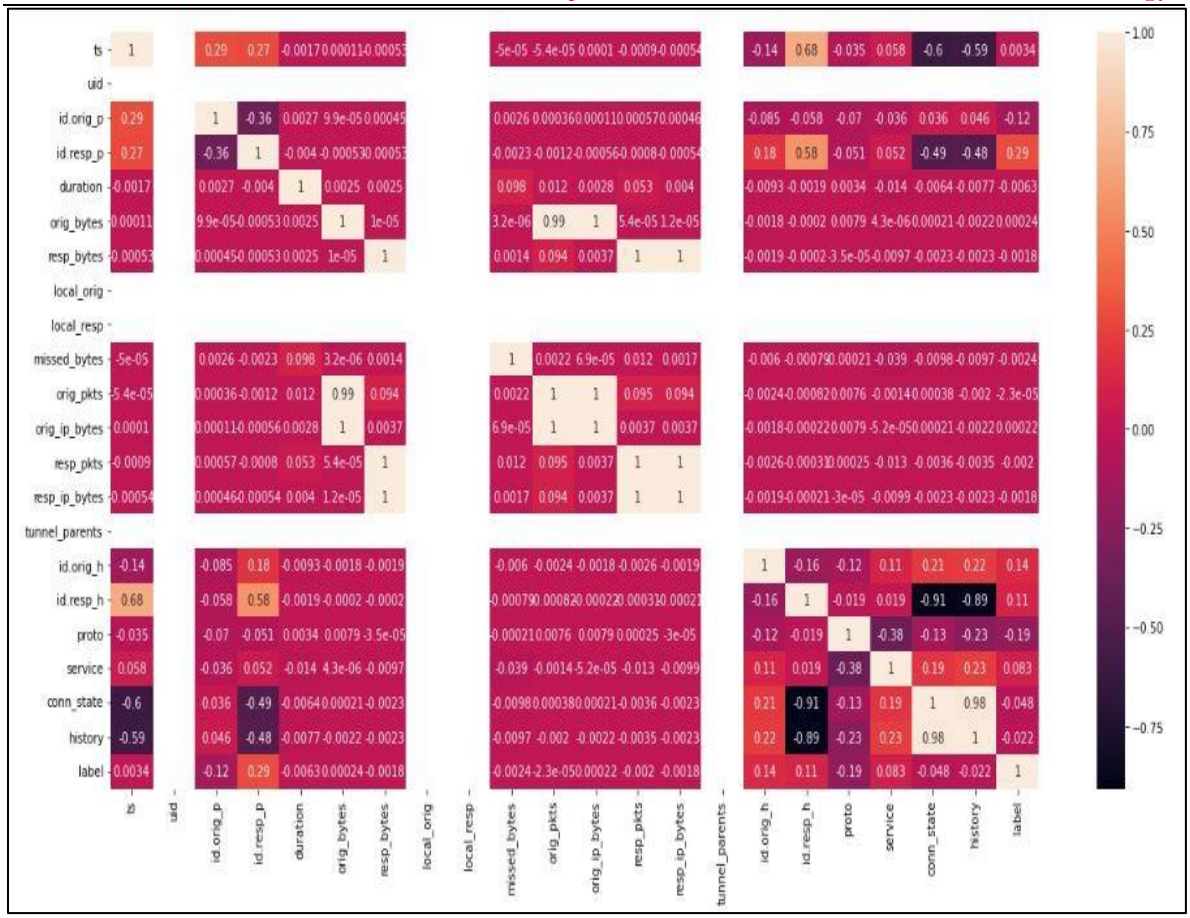

Figure 4. Correlation heat map illustrating the interrelationships and strength of correlations

Correlation Coefficient:

The correlation coefficient was also employed to select the most relevant features from the dataset. Specifically, we used the Pearson correlation coefficient to measure the relationship between two variables. This coefficient quantifies the degree to which two variables are related, with values ranging from -1 to +1. A value of zero indicates no correlation, while a value of +1 signifies a perfect positive correlation. Conversely, a correlation of -1 indicates a perfect negative relationship. A heatmap of the correlation coefficients is presented in **Figure 4**, providing a visual representation of the feature correlations.

Random Forest:

Random forests are one of the most widely used machine learning methods for feature selection. A random forest consists of 400 to 1,200 decision trees, each built using a random subset of both features and observations from the dataset. This randomness helps ensure that the trees are de-correlated, reducing the risk of overfitting, as some trees do not observe all features or data points. Each tree contains a series of binary (yes/no) questions based on one or more features, which split the dataset into two "buckets" at each node. These buckets contain observations that are more similar to each other and different from those in the other bucket. The "purity" of each bucket—how well the observations in it align with the class label—determines the importance of the features used for the split. The features selected based on their high importance in the random forest model are: ts, id.orig_p, id.resp_p, duration, orig_pkts, orig_ip_bytes, resp_ip_bytes, id.orig_h, id.resp_h, conn_state, history, label, and detailed_label.

Result and Discussion:

The experiments were conducted on a system equipped with an Intel Core i7-5600U CPU @ 2.60GHz (quad-core) and 16 GB of RAM. The testing environment included Windows

10, Anaconda Jupyter Notebook, and Python 3.8. To evaluate the model's performance, several metrics were utilized, which are explained in the following sections.

- **Time:** The time taken for an algorithm to execute a machine learning model is an important consideration. In an IoT environment, algorithms that require excessive processing time would be impractical, as they could hinder real-time performance and overall system efficiency.
- **Precision:** Precision is a metric used to evaluate the accuracy of positively identified instances in a model. It is defined as the ratio of correctly identified positive instances to the total number of instances classified as positive, and is given by:

$$\textbf{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}}$$

- **Recall:** Recall is an indicator of the actual number of true positives detected by the model. It is evaluated using the following equation:

$$\textbf{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}}$$

- **F1 Score:** The F1-score calculates the harmonic mean of recall and precision. It is considered a more comprehensive metric because it accounts for both false positives and false negatives. The F1-score is given by:

$$\textbf{F1 Score} = \frac{2 * \text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}}$$

In this context, True Positives refer to instances where the model correctly predicts the positive class, while False Positives indicate cases where the model incorrectly classifies an instance as positive. The details of the experiments, conducted using various combinations of feature selection techniques and classification algorithms, are discussed in the following sections.

Support Vector Machine (SVM):

Support vectors are created using extreme data points to expand the classifier's margin, which in turn enhances classification accuracy. The number of features used to classify the data points determines how many times the SVM algorithm searches for the optimal hyperplane. The hyperplane acts as a decision boundary, separating the data points into different categories on either side. In our experiments, we applied the SVM algorithm after selecting features using each of the feature selection methods individually. The results of the SVM classification are presented in **Table 4**.

Table 4. Experimental results for Support Vector Machine Classification Algorithm

Feature Selection Technique	Accuracy	Precision	Recall	F1 Score
Manual	67.97%	64.18%	67.97%	53.50%
Chi-Square	65.82%	62.12%	65.82%	61.60%
Information Gain	63.22%	60.11%	63.01%	59.20%
Correlation Coefficient	64.29%	61.30%	56.00%	52.10%
Random Forest	65.20%	62.30%	58.47%	53.98%

The results show that the SVM achieved 67.97% accuracy when using manually selected features. In comparison, it classified 65.82%, 63.22%, 64.29%, and 65.20% accurately when using features selected by Chi-Square, Information Gain, Correlation Coefficient, and Random Forest, respectively. For multi-class classification, we employed the SVM with the One-vs-All strategy.

Naïve Bayes:

Bayes' Theorem forms the foundation of the Naive Bayes algorithm, a supervised learning method commonly applied to classification problems. This algorithm makes predictions based on probability, offering a simple yet effective approach for building machine learning models. The results of the Naive Bayes classification algorithm are presented in **Table 5**.

Table 5. Experimental results for the Naive Bayes Classification Algorithm

Feature Selection Technique	Accuracy	Precision	Recall	F1 Score
Manual	65.44%	58.09%	65.45%	53.50%
Chi-Square	85.00%	87.22%	85.00%	79.90%
Information Gain	85.15%	87.29%	85.15%	80.02%
Correlation Coefficient	85.04%	87.33%	85.04%	79.78%
Random Forest	85.15%	87.32%	85.15%	80.06%

The findings show that Naive Bayes achieved the highest accuracy of 85.15% when features were selected using Random Forest or Information Gain. In comparison, it demonstrated classification rates of 65.44%, 85.00%, and 85.04% when applied with Manual Selection, Chi-Square, and Correlation Coefficient methods, respectively.

Decision Trees (DTs):

Supervised machine learning classifiers, such as decision trees (DT), are used for classification tasks where nodes and leaves are connected by branches. The branches represent the decision criteria for classification, while the nodes correspond to the dataset's attributes, outcomes, and leaf nodes. The results of the DT classification algorithm are presented in **Table 6**. The findings indicate that the DT achieved its highest accuracy of 99.99% when feature selection was based on Correlation Coefficient, Chi-Square, Information Gain, and Random Forest techniques. In contrast, when manual feature selection was used, the classification accuracy dropped to 72.37%.

Table 6. Experimental results for the Decision Tree Classification Algorithm

Feature Selection Technique	Accuracy	Precision	Recall	F1 Score
Manual	72.37%	72.31%	72.37%	64.14%
Chi-Square	99.99%	99.99%	99.99%	99.99%
Information Gain	99.99%	99.99%	99.99%	99.99%
Correlation Coefficient	99.99%	99.99%	99.99%	99.99%
Random Forest	99.99%	99.99%	99.99%	99.99%

Results Comparison:

We applied multiple feature selection algorithms, each of which returned slightly different sets of features. As a result, each classification algorithm yielded different outcomes depending on the features used. **Figure 5** presents a comparison of the accuracy achieved by each classification algorithm after applying the various feature selection methods.

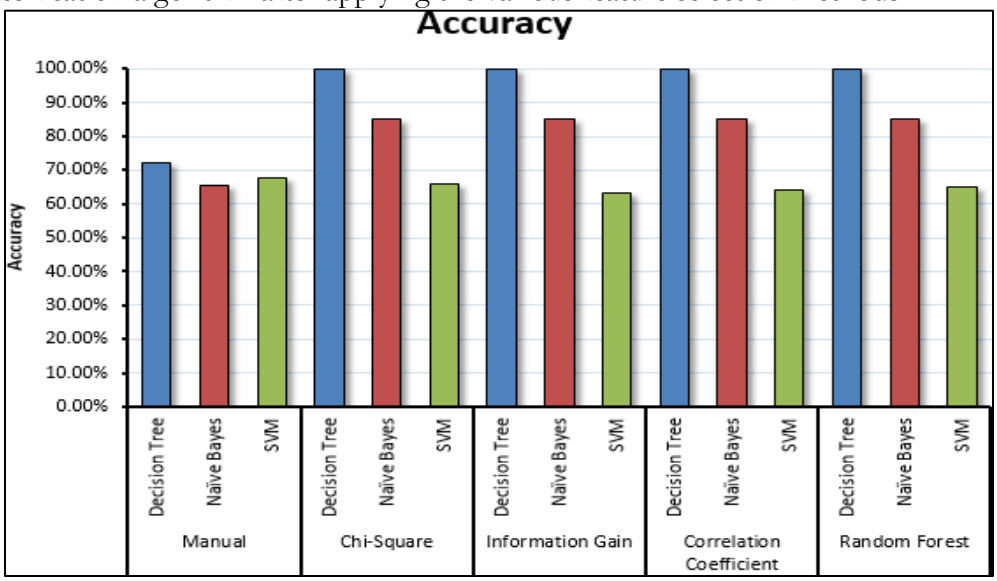

Figure 5. Accuracy scores of different classification algorithms evaluated in the proposed methodology for the classification task.

The results from our experimental analysis on the IoT-23 dataset provide valuable insights into the effectiveness and efficiency of different feature reduction techniques and classification algorithms for anomaly detection in IoT networks. Notably, the Decision Tree (DT) algorithm demonstrated exceptional performance, achieving an accuracy rate of 99.99%. Furthermore, it exhibited low computational cost, taking only about 5.2 seconds to execute with the Correlation Coefficient method. These factors highlight the efficacy of the DT algorithm in detecting anomalies within IoT networks. The appeal of using this method lies in its ability to capture complex decision boundaries while also providing interpretability, making it a strong choice for identifying abnormalities in IoT environments.

While Figure 6 shows the comparison with respect to time cost achieved by each classification algorithm.

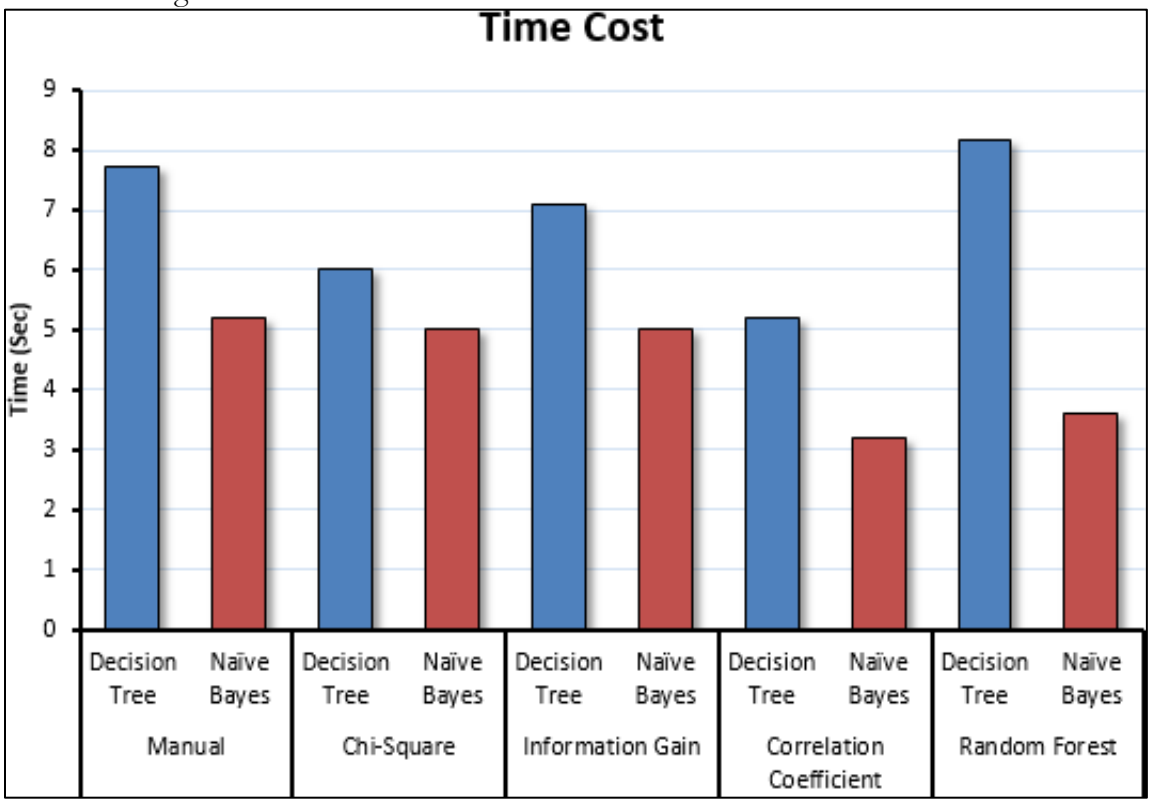

Figure 6. Time cost (in seconds) of executing different classification algorithms.

In summary, the SVM algorithm achieved its highest accuracy of 67.97%, with a significant time cost of 4666 seconds when using manual feature selection. The Naive Bayes algorithm reached a maximum accuracy of 85.15%, with a minimal time cost of 3.6 seconds when utilizing Random Forest for feature selection. For the Decision Tree (DT) algorithm, the highest accuracy was 99.99%, with a time cost of just 5.2 seconds when using the Correlation Coefficient method for feature selection. Overall, the DT algorithm achieved 99.99% accuracy in 5.2 seconds, the Naive Bayes algorithm achieved 85.04% accuracy in 3.2 seconds, and the SVM algorithm reached 67.97% accuracy in 193.6 seconds.

Conclusion and Future Work:

The primary goal of this research was to enhance the security of the Internet of Things (IoT), with a particular emphasis on smart homes and their integration into IoT-enabled cloud computing environments. Throughout our investigation, we successfully identified key challenges and proposed practical solutions to protect data, detect anomalies, and prevent unauthorized access within IoT ecosystems. Our findings highlight the critical need for incorporating Intrusion Detection Systems (IDS) into smart homes powered by IoT technologies.

We evaluated various machine learning algorithms using the IoT-23 dataset to assess their performance. Among the algorithms tested, the Decision Tree (DT) algorithm demonstrated the highest levels of accuracy and efficiency. In contrast, the Naive Bayes algorithm delivered comparatively less favorable results. To further advance our research, we recommend exploring datasets with a larger number of features, which will allow for a more comprehensive evaluation of the proposed approaches.

In conclusion, our research underscores the importance of implementing robust security protocols, such as IDS, to ensure the security and integrity of IoT-enabled cloud computing environments. By addressing the dynamic nature of security challenges within IoT networks, we can improve the performance, functionality, and overall capabilities of IoT applications. We look forward to future research efforts that will explore these aspects further and contribute to advancing IoT security in the context of IoT-enabled cloud computing environments.

Author's Contribution: A.H.F worked on the methodology and conceptualization of the proposed method. R.A. performed the formal analysis and provided the original draft. S. K. reviewed, made final draft, and supervised the work.

Conflict of Interest. Authors had no conflict of interest for publishing this manuscript in IJIST.

References:

[1] Wang, H.; Ning, H.; Lin, Y.; Wang, W.; Dhelim, S.; Farha, F.; Ding, J.; Daneshmand, M. A Survey on the Metaverse: The State-of-the-Art, Technologies, Applications, and Challenges. IEEE Internet of Things Journal **2023**, Vol 10, pp. 14671–14688.

[2] Farooqi, A.H.; Akhtar, S.; Rahman, H.; Sadiq, T.; Abbass, W. Enhancing Network Intrusion Detection Using an Ensemble Voting Classifier for Internet of Things. Sensors **2024**, 24, 127. https://doi.org/10.3390/s24010127

[3] Wang, Y.; Su, Z.; Zhang, N.; Xing, R.; Liu, D.; Luan, T.H.; Shen, X. A Survey on Metaverse: Fundamentals, Security, and Privacy. IEEE Communications Surveys Tutorials **2023**, Vol 25, pp. 319–352. http://doi:10.1109/COMST.2022.3202047

[4] Kang, G.; Koo, J.; Kim, Y.G. Security and Privacy Requirements for the Metaverse: A Metaverse Applications Perspective. IEEE Communications Magazine **2023**, pp. 1–7.

[5] Dhiman, P.; Saini, N.; Gulzar, Y.; Turaev, S.; Kaur, A.; Nisa, K.U.; Hamid, Y. A Review and Comparative Analysis of Relevant Approaches of Zero Trust Network Model. Sensors **2024**, 24, 1328. https://doi.org/10.3390/s24041328

[8] Alam, T. Blockchain-Based Internet of Things: Review, Current Trends, Applications, and Future Challenges. Computers **2023**, 12. http://doi:10.3390/computers12010006

[9] Shwe, T.; Aritsugi, M. Optimizing Data Processing: A Comparative Study of Big Data Platforms in Edge, Fog, and Cloud Layers. Applied Sciences. **2024**, 14, 452. https://doi.org/10.3390/app14010452

[10] Sattari, F.; Farooqi, A.H.; Qadir, Z.; Raza, B.; Nazari, H.; Almutiry, M. A Hybrid Deep Learning Approach for Bottleneck Detection in IoT. IEEE Access **2022**, Vol. 10, pp. 77039–77053. https://10.1109/ACCESS.2022.3188635

[11] Sebastian Garcia, Agustin Parmisano, "IoT-23: A labeled dataset with malicious and benign IoT network traffic". Zenodo **2020**.

[12] Alani, M.M.; Miri, A. Towards an Explainable Universal Feature Set for IoT Intrusion Detection. Sensors **2022**, Vol. 22. https://doi:10.3390/s22155690

[13] Li, J., Othman, M.S., Chen, H. et al. Optimizing IoT intrusion detection system: feature selection versus feature extraction in machine learning. J Big Data **2024**, Vo. 11, 36. https://doi.org/10.1186/s40537-024-00892-y

[14] Khan, F.A., Farooqi, A.H. & Derhab, A. A comprehensive security analysis of LEACH++ clustering protocol for wireless sensor networks. Journal of Supercomputing **2019**, 75, 2221–2242. https://doi.org/10.1007/s11227-018-2680-3

[15] Khan R, Tariq N, Ashraf M, Khan FA, Shafi S, Ali A. FL-DSFA: Securing RPL-Based IoT Networks against Selective Forwarding Attacks Using Federated Learning. Sensors. **2024**; 24(17):5834. https://doi.org/10.3390/s24175834

[16] Lin, J.; Yu, W.; Zhang, N.; Yang, X.; Zhang, H.; Zhao, W. A Survey on Internet of Things: Architecture, Enabling Technologies, Security and Privacy, and Applications. IEEE Internet of Things Journal **2017**, 4, 1125–1142. https://doi:10.1109/JIOT.2017.2683200

[17] Panza, M.A.; Pota, M.; Esposito, M. Anomaly Detection Methods for Industrial Applications: A Comparative Study. Electronics **2023**, Vol 12.

[18] Park, C.; Lee, J.; Kim, Y.; Park, J.G.; Kim, H.; Hong, D. An Enhanced AI-Based Network Intrusion Detection System Using Generative Adversarial Networks. IEEE Internet of Things Journal **2023**, Vol. 10, 2330–2345. https://doi:10.1109/JIOT.2022.3211346

[19] Benkhelifa, E.; Welsh, T.; Hamouda, W. A critical review of practices and challenges in intrusion detection systems for IoT: Toward universal and resilient systems. IEEE Communications Surveys & Tutorials **2018**, Vol. 20, 3496–3509.

[20] Pajouh, H.; Javidan, R.; Khayami, R.; Dehghantanha, A.; Choo, K. A Two-Layer Dimension Reduction and Two-Tier Classification Model for Anomaly-Based Intrusion Detection in IoT Backbone Networks. IEEE Transactions on Emerging Topics in Computing **2019**, 7, 314–323. https://doi:10.1109/TETC.2016.2633228

[21] Rabhi, S.; Abbes, T.; Zarai, F. "IoT botnet detection using deep learning." 2023 International Wireless Communications and Mobile Computing, **2023**, pp. 1107–1111.

[22] Caldas Filho, F.L.; Soares, S.C.M.; Oroski, E.; de Oliveira Albuquerque, R.; da Mata, R.Z.A.; de Mendonça, F.L.L.; de Sousa Júnior, R.T. Botnet Detection and Mitigation Model for IoT Networks Using Federated Learning. Sensors **2023**, Vol. 23. pp. 1-21.

[23] Alosaimi, S.; Almutairi, S.M. An Intrusion Detection System Using BoT-IoT. Applied Sciences **2023**, Vol. 13, 9: 5427. https://doi:10.3390/app13095427

[24] Gromov, M.; Arnold, D.; Saniie, J. Utilizing Computer Vision Algorithms to Detect and Classify Cyberattacks in IoT Environments in Real-Time. 2023 IEEE International Conference on Electro Information Technology, **2023**, pp. 300–303.

[25] Li, J.; Zhao Z.; Li, R.; Zhang, H., "AI-Based Two-Stage Intrusion Detection for Software Defined IoT Networks," in IEEE Internet of Things Journal **2019**, Vol. 6, 2, pp. 2093-2102. https://doi:10.1109/JIOT.2018.2883344

[26] Latif, S.; Huma, Z.e.; Jamal, S.S.; Ahmed, F.; Ahmad, J.; Zahid, A.; Dashtipour, K.; Aftab, M.U.; Ahmad, M.; Abbasi, Q.H. Intrusion Detection Framework for the Internet of Things Using a Dense Random Neural Network. IEEE Transactions on Industrial Informatics **2022**, Vol. 18, 6435–6444. https://doi:10.1109/TII.2021.3130248

[27] Kamaldeep; Malik, M.; Dutta, M.; Granjal, J. IoT-Sentry: A Cross-Layer-Based Intrusion Detection System in Standardized Internet of Things. IEEE Sensors Journal **2021**, Vol. 21, 24, pp. 28066-28076. https://doi:10.1109/JSEN.2021.3124886

[28] Jeelani, F.; Rai, D.S.; Maithani, A.; Gupta, S. The Detection of IoT Botnet using Machine Learning on IoT-23 Dataset. 2nd International Conference on Innovative Practices in Technology and Management, Gautam Buddha Nagar, India, **2022**, pp. 634-639, https://doi:10.1109/ICIPTM54933.2022.9754187