ENHANCING DATA PROCESSING EFFICIENCY IN IOT INTRUSION DETECTION SYSTEMS THROUGH FEATURE REDUCTION

Quang-Truong Can[1], Trung-Ninh Bui[1], Thai-Mai Dinh Thi[1,*]

DOI: http://doi.org/ 10.57001/huih5804.2025.350

ABSTRACT

The rapid growth of IoT devices has improved connectivity but has also made them vulnerable to attacks due to limited resources. The Intrusion Detection System (IDS) is one of the effective ways, where machine learning-based defense mechanisms are used for early detection. Due to the limitations of resources and computing in IoT devices, optimizing IDS plays a crucial role. The optimization process includes many phases, in which the preprocessing phase helps reduce the dimensions of features and speed up input processing. In this paper, we utilized the Edge-IIoT dataset to evaluate two feature reduction techniques, including feature selection (FS) and feature extraction (FE) for machine learning models, that reduce the complexity of machine learning. We used Pearson Correlation (PCC) and Principal Component Analysis (PCA) algorithms for FS and FE, respectively. As a result, we found that FE is better than FS, with small features and stability, but it takes more time for training compared to the other one. In contrast, FS is better than FE when increasing the number of features, which results in outperformance in accuracy and requires less time to reduce features.

Keywords: *Machine learning, feature reduction, IoT, Cybersecurity, IDS, Edge-IIioT.*

[1]Faculty of Electronisc and Telecommunications, University of Engineering and Technology, Vietnam National University, Hanoi, Vietnam

[*]Email: dttmai@vnu.edu.vn

Received: 26/6/2025
Revised: 15/8/2025
Accepted: 28/9/2025

1. INTRODUCTION

The Internet of Things [11] (IoT) has emerged in recent years, IoT devices have rapidly evolved into a wide range of types, including sensors, smart cameras, smart televisions, and other household devices. These devices are interconnected to create a massive network that exchanges trillions of data points. Indeed, most IoT devices have limited resources so that this vulnerability makes them susceptible to security issues, such as DDoS, SQL injections, malware attacks, etc.

Many solutions have been explored and implemented to protect IoT systems. One such solution is a Network Intrusion Detection System (NIDS) [12]. NIDS can detect abnormal intrusions in IoT networks and prevent malicious access early. NIDS relies on various methods, such as statistics-based, pattern-based, etc. Among them, Machine Learning and Deep Learning stand out as state-of-the-art (SOTA) approaches. Most machine learning models are based on available datasets. However, a common weakness of these datasets is the lack of proper data cleaning, which results in redundant information and features. Many researchers have investigated this problem extensively and proposed various methods that can generally be categorized into two types: feature selection (FS) techniques and feature extraction (FE) techniques. In fact, most studies focus only on FS or FE techniques individually and do not evaluate them comprehensively within a specific case study.

For evaluating and comparing purposes, we will implement a framework in which FE and FS will be compared by using a the IIoT traffic dataset, named Edge-IIoT dataset [6]. Overall, our findings show that FE performs better than FS when working with a smaller number of features and remains stable as the number of features increases. However, the FE requires more time for the training process. In contrast, the FS outperforms FE in terms of accuracy when using a larger number of features and requires less training time. These evaluations are conducted using well-known machine learning models, including K-Nearest Neighbors (KNN), Decision Tree (DT), Random Forest (RF), Light Gradient-Boosting Machine (LightGBM) and Multi Layer

Perceptron (MLP). Performance is assessed using common metrics such as accuracy, precision, F1 score, and recall. The key contributions of this research are as follows:

- Deploy a framework for data prepocessing on Edge-IIoT dataset.

- Comparing and evaluating two feature reduction methods including FS and FE techniques.

The structure of this paper includes the following sections: Section 2. Related Works will investigate similar topics, Section 3. Methodology presents the methods used for evaluation; Section 4. Implementation and Evaluation details the configurations, model execution, and performance comparison of the two approaches; and finally, Section 5. Conclusion will summarize the findings and discuss future work.

2. RELATED WORKS

In terms of feature selection (FS) techniques, various methods have been employed to enhance the performance of intrusion detection systems (IDS). In paper [8], the authors utilized correlation and mutual information to select optimal features for addressing the challenge of continuous input features and discrete target values. Their approach achieved a high detection accuracy of 99.9% for DDoS attacks. Similarly, in paper [9], the authors analyzed the UNSW-NB15 Dataset [13] and applied the XGBoost-based feature selection method. This method led to an accuracy improvement, from 88.13% to 90.85%, for the binary classification scheme. Moreover, paper [1] explored the application of decision tree classifiers on reduced feature sets for building an IDS. The results indicated that selecting reduced attributes through a novel feature selection system contributed to better performance and a computationally efficient IDS. These studies highlight the importance of effective feature selection in developing robust and efficient IDS. However, in the context of the Internet of Things (IoT), resource constraints such as limited processing power, memory, and energy pose significant challenges for directly applying these feature selection methods.

In terms of feature extraction (FE) techniques, methods like Principal Component Analysis (PCA), Linear Discriminant Analysis (LDA), and neural network-based Autoencoders (AE) have been widely used to reduce dimensionality in Network Intrusion Detection Systems (NIDS). In [16], the authors employed PCA for feature reduction, achieving a performance time of 3.24 minutes, an accuracy of 96.78%, and an error rate of 0.21%. Similarly, [4] reported high performance with PCA while maintaining lower feature dimensions. In [2], Autoencoders reduced features to a 3-dimensional latent space (90% compression), with minimal impact on detection accuracy. LDA has also been applied to significantly reduce computational complexity in NIDS, as shown in [14]. Furthermore, hybrid approaches that combine multiple feature extraction methods, such as the one described in [3], have demonstrated greater robustness compared to single-method techniques.

While existing studies focus on FS or FE techniques, they often lack an analysis of their impact on model performance, use outdated datasets, and overlook resource constraints in IoT scenarios. Our research addresses these gaps by comparing Pearson correlation and PCA as lightweight methods on the Edge-IIoT dataset. In this study, we focus on well-known machine learning models such as K-Nearest Neighbors (KNN), Decision Trees (DT), and Random Forest (RF). The next section will detail this approach.

3. METHODOLOGY

In this section, we will evaluate the FS and FE techniques using the framework as shown in Figure 1. These methods will be evaluated and compared based on multi-class classification model performance. The pipeline process of the framework is divided into three sections:

Phase 1: Data pre-processing

In this phase, we will clean the dataset by removing missing values, eliminating duplicates, dropping redundant columns, and encoding for non-numerical data as described in [6].

Phase 2: Feature reduction

In this phase, the FS and FE techniques, including Pearson's Correlation Coefficient (PCC) and PCA will be applied to reduce the feature number. This step provides in depth insight into the impact of each method on the Edge-IIoT dataset.

Phase 3: Classification Modeling

After going through the feature reduction phase, the data will be split into training and test sets. We will use several well-known models, such as KNN, DT, RF, LightGBM and MLP to evaluate the performance of the feature reduction techniques.

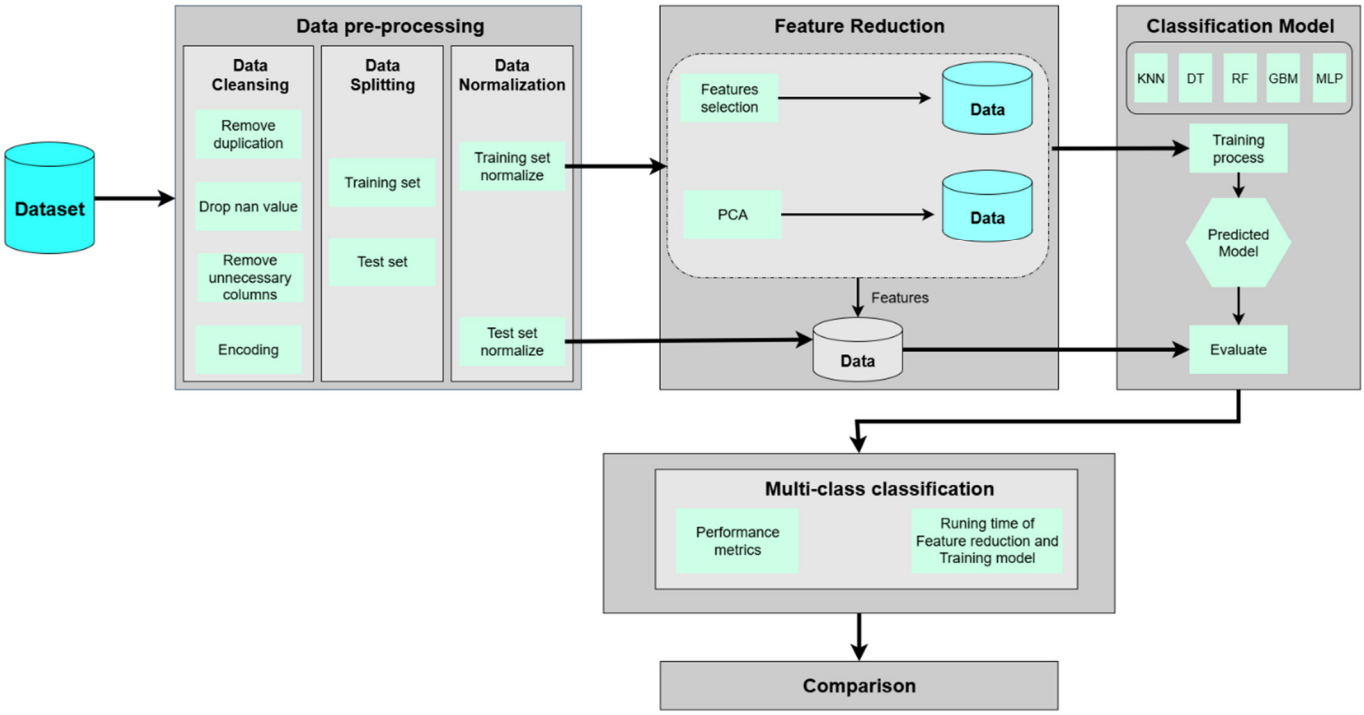

Figure 1. Framework for comparison Feature Selection and Feature Extraction techniques

Feature Selection

The Pearson's Correlation Coefficient (PCC) method is a straightforward approach for assessing linear correlations between features, allowing us to evaluate feature interdependencies. By identifying features with high correlation coefficients, we aim to remove redundant features that contribute little additional information to the model. This process relies on a correlation matrix. To calculate PCC between two features, we use [17]:

$$\mathrm{PCC}(f_1, f_2) = \frac{\mathrm{cov}(f_1, f_2)}{\sigma_{f_1} \times \sigma_{f_2}} \quad (1)$$

In (1), cov and σ represent the covariance and standard deviation, respectively. To select important features, we will choose many threshold values increasingly. As explained above, features that have high correlation values will be removed because they contribute less information than the other ones. The PCC can be positive or negative values, so we will choose thresholds that range values from negative to positive threshold symmetrically.

Feature Extraction

Among various algorithms for feature extraction, PCA [7], Autoencoders (AE) [5], and Linear Discriminant Analysis (LDA) [15] are popular choices. However, PCA stands out as it adapts well to IoT resource constraints and is computationally efficient, making it faster to calculate compared to other methods.

With matrix $\mathbf{X}$ representing a dataset and $\hat{\mathbf{X}}$ as the normalized form of $\mathbf{X}$, the correlation matrix is calculated by the following formulas

$$\mathbf{S} = \frac{1}{N}\,\hat{\mathbf{X}}\hat{\mathbf{X}}^{T} \quad (2)$$

In (2), N denoted number of elements in $\mathbf{X}$ and $\hat{\mathbf{X}}$ is a normalized matrix of $\hat{\mathbf{x}}_n$ vectors calculated by following formulas

$$\bar{\mathbf{x}} = \frac{1}{N}\sum \mathbf{x_n} \quad (3)$$

Here, $\bar{\mathbf{x}}$ represents the mean of N vectors $\mathbf{x}_n$, and $\hat{\mathbf{x}}_n$ is

$$\hat{\mathbf{x}}_n = \mathbf{x}_n - \bar{\mathbf{x}} \quad (4)$$

In (4), $\hat{\mathbf{x}}$ is the normalization of $\mathbf{x}$ by subtracting the mean $\bar{\mathbf{x}}$.

From the correlation matrix, we compute the eigenvectors and eigenvalues of $\mathbf{S}$. By sorting the eigenvalues in decreasing order, we obtain the matrix $\mathbf{U}$. From $\mathbf{U}$, we select $\mathbf{K}$ eigenvectors corresponding to the largest eigenvalues to form the new dimensions in the transformed space. The matrix $\mathbf{Z}$ is then computed as the projection of $\mathbf{X}$ onto this new space with $\mathbf{K}$ dimensions, preserving the most significant information

$$\mathbf{Z} = \mathbf{U}_{\mathbf{K}}^{\mathbf{T}}\hat{\mathbf{X}} \quad (5)$$

Here, the $\mathbf{X}$ dataset is converted to $\mathbf{Z}$ in a new space and the remaining $\mathbf{K}$ features.

4. IMPLEMENTATION AND EVALUATION

4.1. Implementation

Setup

The framework in Figure 1 will be implemented on Kaggle, leveraging its powerful processing and collaborative tools for efficient execution and evaluation. We will assess model performance in terms of Accuracy, Precision, F1-Score, and Recall. We measure the time taken by Feature Selection and Feature Extraction processes to evaluate their impact.

We use Edge-IIoT dataset [6] to evaluate two features reduction techniques. The Edge-IIoT dataset is a comprehensive collection from over ten IoT device types, integrated with various technology stacks, including Cloud, Edge, Fog Computing, SDN, Blockchain, and multiple protocols. It captures fourteen types of attacks, grouped into five main categories: DoS/DDoS, Information Gathering, Man-in-the-Middle, Injection, and Malware, making it a realistic resource for cybersecurity research, as shown in Table 1. This dataset includes two folders for evaluating ML and DL models. In this study, we focus on the ML dataset, a smaller subset of the DL dataset, to reduce analysis time and streamline evaluation.

Table 1. Edge-IIoT summary of records [6]

IoT traffic	Class	Records	Total
Normal	Normal	11,223,940	11,223,940
Attack	Backdoor attack	24,862	9,728,708
	DDoS_HTTP attack	229,022	
	DDoS_ICMP attack	2,914,354	
	DDoS_TCP attack	2,020,120	
	DDoS_UDP attack	3,201,626	
	Fingerprinting attack	1,001	
	MITM attack	1,229	
	Password attack	1,053,385	
	Port_Scanning attack	22,564	
	Ransomware attack	10,925	
	SQL_injection attack	51,203	
	Uploading attack	37,636	
	Vulnerability_scanner attack	145,869	
	XSS attack	15,915	
Total			2,095,2648

We utilized a diverse set of five machine learning models in our study, including DT, KNN, RF, LightGBM, and MLP. The specific configurations and settings for each model after fine-turning, including hyperparameters, are comprehensively summarized in Table 2.

Table 2. Configuration Model

Model	Parameters
KNN	n_neighbors=4
DT	random_state=0, max_depth=14
RF	random_state=0, max_depth=14
LightGBM	learning_rate=0.02, max_depth=10, num_leaves=50
MLP	hidden_layer_sizes=64, batch_size=512, random_state=0, max_iter=100

Feature selected by threshold correlation

The features selected by the Feature Selection method will be chosen based on a specific threshold. Figure 2 presents the Pearson Correlation Coefficient (PCC) of each feature, excluding those with NaN values, as discussed in Section 3. This visualization helps identify the features that meet the correlation threshold criteria and are thus suitable for selection, ensuring that only the most relevant features are retained. Detail number of selected features and chosen thresholds in Table 3.

We can see that features of TCP and HTTP protocols contribute a lot of information based on PCC value, especially the HTTP fields such as version and status. Most of the features of the MQTT protocol obtain high PCC values, which do not contribute good information for the model.

Table 3. Selected features by Feature Selection

Range	No. Selected Features
[−0.01, 0.01]	17
[−0.015, 0.015]	23
[−0.03, 0.03]	53
[−0.05, 0.05]	67
NaN	74

Feature selected by PCA

In this approach, we extracted the same number of features as in the Feature Selection method for a meaningful comparison. After applying PCA to the Edge-IIoT dataset, we generated a chart illustrating the new features in the reduced dimensional space, as shown in Figures 3 and 4.

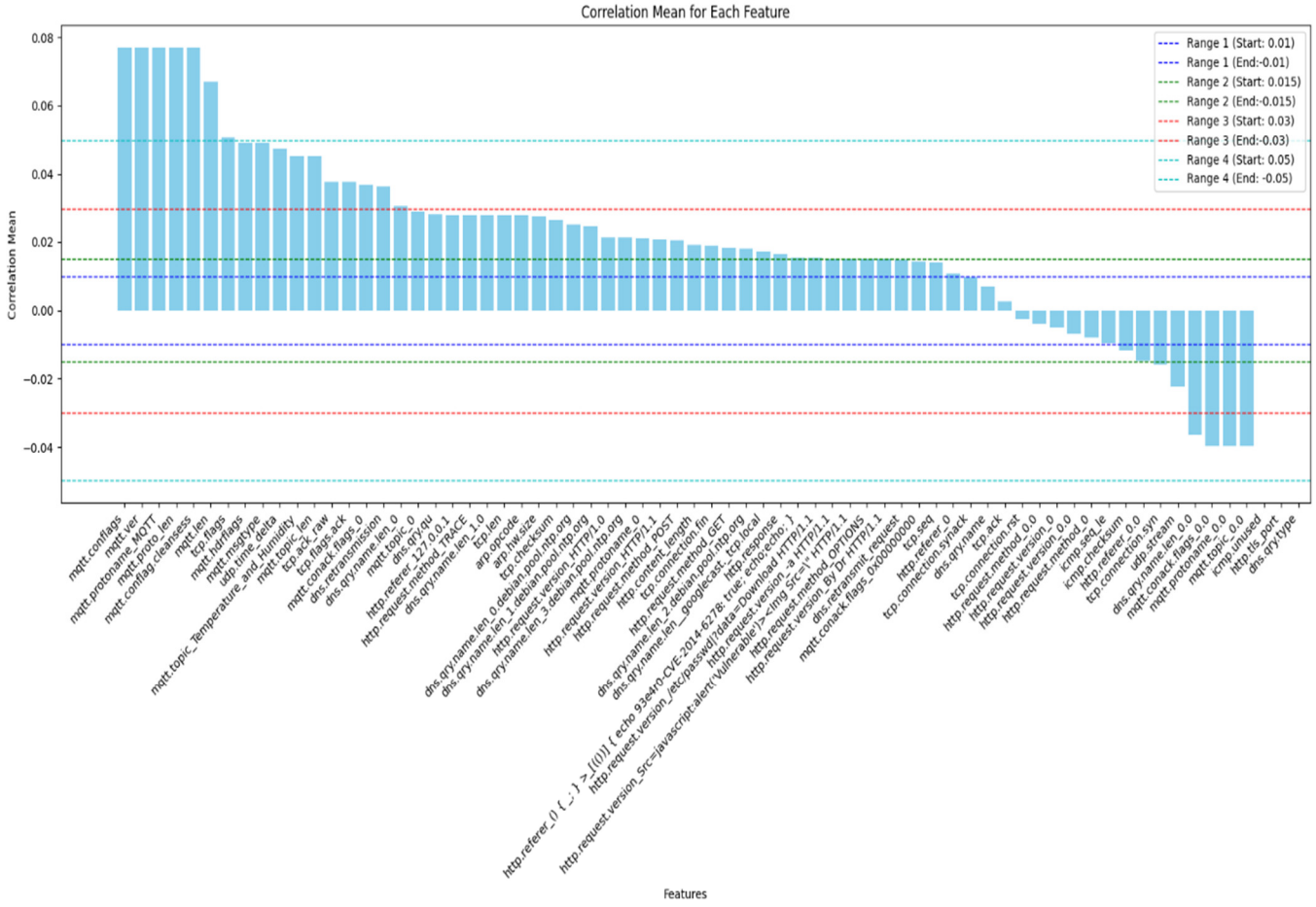

Figure 2. Correlation Mean for Each Feature

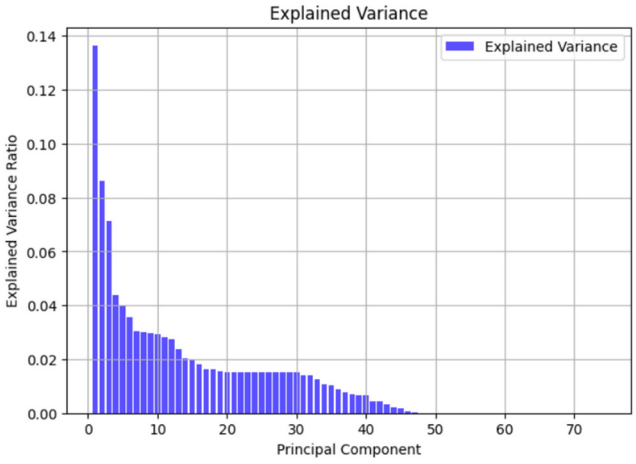

Figure 3. Explained Variance

In Figure 3, the new features are sorted by decreasing explained variance ratio. As observed, the most important features appear 50 first, as they contribute more significant information compared to the others. Mapping to the cumulative explained variance ratio shown in Figure 4, in 50 first features, the cumulative explained variance obtained 99% information about the dataset.

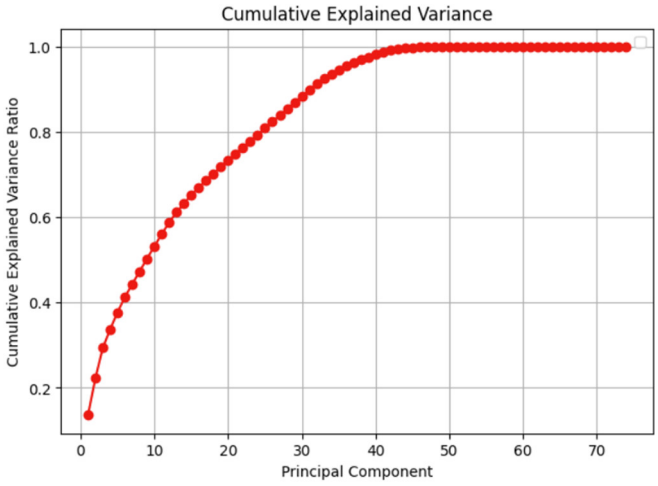

Figure 4. Culmulative Explain Variance

4.2. Evaluation

After fitting two feature reduction methods on five models, we summary results in Table 4 to Table 7. The Acc, Pre, Re and F1 stand for Accuracy, Precision, Recall and F1-Score, respectively. We gradually increased the number of features to evaluate the impact and performance of each dimensionality reduction method. As shown in Table 4, with 17 features, Feature Extraction

(FE) demonstrates significantly better performance compared to Feature Selection (FS). The MLP model achieved the best results, with an accuracy of 81.58%, outperforming FS methods by over 40%. Metrics such as Precision, Recall, and F1-Score were also notably higher, reflecting FE's ability to preserve critical information for classification tasks. Additionally, FE reduced the dimensionality faster than FS, showcasing its efficiency in preprocessing. However, FS exhibited quicker model training times compared to FE, making FS more suitable for scenarios where training time is a critical factor.

Table 4. Results with 17 features

	Acc (%)	Pre (%)	Re (%)	F1 (%)	Feature Reduction Time (s)	Training Time (s)
Feature Selection						
DT	50.85	67.4	48.68	46.55	2.43	0.18
RF	51.57	72.94	49.56	48.15		7.89
KNN	63.07	70.63	62.56	61.84		0.014
LightGBM	75.7	79.66	75.7	75.85		18.76
MLP	41.95	39.36	41.95	32.15		51.12
Feature Extraction						
DT	81.18	82.67	78.45	78.64	1.65	3.43
RF	80.8	81.88	77.93	78.48		69.21
KNN	79.81	77.35	75.97	76.42		0.015
LightGBM	81.55	79.9	79.81	79.73		0.02
MLP	81.58	84.21	81.58	81.58		38.65

Table 5. Result with 23 features

	Acc (%)	Pre (%)	Re (%)	F1 (%)	Feature Reduction Time (s)	Training Time (s)
Feature Selection						
DT	79.38	85.23	78.45	78.76	2.43	0.23
RF	79.47	84.98	78.55	78.86		8.52
KNN	80.42	80.98	78.01	78.26		0.014
LightGBM	84.57	87.37	84.57	84.22		19.36
MLP	53.05	64.43	53.05	48.22		55.15
Feature Extraction						
DT	81.57	85.58	78.51	78.23	1.98	4.65
RF	81.25	82.3	78.79	79.06		67.98
KNN	79.97	78.28	77.44	77.72		0.015
LightGBM	81.62	83.96	81.61	81.58		18.08
MLP	80.99	86.33	80.99	80.41		32.72

When the number of features increased to 23, as shown in Table 5, FS demonstrated remarkable improvements in model performance compared to its results with 17 features. Notably, the performance metrics of all models increased with FS as the number of features grew. This result highlights FS's ability to better utilize additional features for classification as the feature set expands. In contrast, FE showed stability in these metrics, indicating that its performance gains plateaued with the addition of more features. Regarding dimensionality reduction time, FE remained about 1 second faster than FS, reaffirming its preprocessing efficiency. However, the training time for FE was considerably longer, with RF requiring an average of 68 seconds for training.

As the number of features increased to 53, 67, and eventually 74 (full of features), corresponding to Tables 6, 7, and 8, FS performance improved significantly. The accuracy of FS models reached up to 94% (LightGBM), with Precision peaking at approximately 93%. Recall and F1-Score also saw substantial improvements, reaching nearly 90%. This demonstrates FS's ability to leverage a larger feature set effectively, particularly with LightGBM, which consistently performed best among FS models. Conversely, FE metrics remained relatively stable, showing minimal improvement even compared to scenarios with fewer features, such as 17 or 23, suggesting diminishing returns for FE as the feature set grows. While FE continued to have faster dimensionality reduction times than FS, this advantage was outweighed by its significantly longer training times, particularly for RF, which required over 108 seconds with the full feature set. These findings highlight FS as a more practical choice when working with larger feature sets, particularly in time-sensitive applications, while FE remains advantageous for small feature subsets where its efficiency and performance benefits are more pronounced.

Table 6. Results with 53 features

	Acc (%)	Pre (%)	Re (%)	F1 (%)	Feature Reduction Time (s)	Training Time (s)
Feature Selection						
DT	89.38	92.31	86.43	87.66	2.51	0.87
RF	89.64	91.41	87.35	88.27		13.95
KNN	77.57	75.78	74.91	75.16		0.017
LightGBM	91.75	92.88	91.75	91.7		24.13
MLP	77.97	80.62	77.97	77.75		63.25

Feature Extraction						
DT	81.85	84.49	79.4	79.47	2.93	8.94
RF	81.86	83.73	79.45	79.69		107.63
KNN	79.81	78.26	77.3	77.62		0.017
LightGBM	81.81	84.07	81.81	81.81		36.78
MLP	81.51	84.7	81.51	81.55		34.78

Table 7. Results with 67 features

	Acc (%)	**Pre (%)**	**Re (%)**	**F1 (%)**	**Feature Reduction Time (s)**	**Training Time (s)**
Feature Selection						
DT	91.4	93.71	89.22	90.18	2.61	1.11
RF	91.73	92.41	89.59	90.42		15.89
KNN	79.03	77.54	76.54	76.89		0.018
LightGBM	93.68	94.36	93.68	93.7		25.41
MLP	81.58	84.21	81.58	81.58		64.97
Feature Extraction						
DT	81.85	84.54	79.39	79.48	0.81	9.18
RF	81.91	83.85	79.55	79.8		108.07
KNN	79.81	78.26	77.31	77.63		0.018
LightGBM	82.84	84.74	82.84	82.86		44.81
MLP	81.67	86.94	81.67	80.96		37.38

Table 8. Results with full of features

	Acc (%)	**Pre (%)**	**Re (%)**	**F1 (%)**	**Feature Reduction Time (s)**	**Training Time (s)**
Feature Selection						
DT	91.41	93.65	89.12	90.13	2.64	1.13
RF	91.08	91.77	88.98	89.84		18.25
KNN	79.81	78.25	77.3	77.62		0.019
LightGBM	93.72	94.38	93.72	93.74		24.55
MLP	81.24	85.32	81.24	80.72		67.26
Feature Extraction						
DT	81.85	84.54	79.43	79.51	0.86	9.42
RF	81.88	83.79	79.5	79.74		106.1
KNN	79.81	78.26	77.31	77.63		0.019
LightGBM	82.83	84.73	82.83	82.86		48.95
MLP	81.5	84.01	81.5	81.5		38.17

When using FS with a small number of features, critical information is often discarded, leading to poorer performance metrics. In contrast, with FE, specifically PCA, the transformation ensures that key information is concentrated in the first few components. As shown in Figures 3 and 4, most of the information is captured within the first 47 components, with the highest concentration in the first three components. This explains the stability in performance metrics for FE as the number of features increases, while FS exhibits significant variability due to the inclusion of more complete information.

Furthermore, it is important to note that PCA transforms the data into a new dimensional space, which can result in some loss of information. This may account for the accuracy plateauing at around 81% in our study. These observations align with the findings reported by the authors in [10] demonstrating the robustness of their conclusions when applied to the Edge-IIoT dataset in our experiments. The summarized observations and insights when analyzing with Edge-IIoT dataset are presented in the comparative Table 9.

Table 9. Summary in comparison between FS and FE

No	**Content**	**FS**	**FE**
1	Higher accuracy when a small number of features		✓
2	Higher accuracy when increasing features	✓	
3	Lower time for reduction features		✓
4	Lower time for training model	✓	
5	Less sensitive to the number of selected/extracted features		✓
6	Greater potential for performance improvement with fewer features.		✓
7	Effectiveness in High-Dimensional Data		✓
8	Effective in Computational Complexity During Reduction	✓	
9	Suitability for Real-Time Systems	✓	

5. CONCLUSION

This study evaluated FS and FE techniques using the Edge-IIoT dataset to assess their effectiveness in an IoT Intrusion Detection System. Our findings reveal that with fewer features, FE achieves higher accuracy than FS, which tends to lose accuracy in such cases. Conversely, FS performs better with larger feature sets, while FE accuracy drops under the same conditions. Notably, FE maintains stability as the number of features increases, demonstrating its robustness compared to FS.

We also observed that training time increases significantly when using PCA for FE compared to using

PCC-based Feature Selection, highlighting the trade-off between computational efficiency and model stability.

Future work will explore hybrid methods that combine FS and FE, as well as investigate new feature reduction algorithms designed to better accommodate the computational limitations of IoT devices. This approach aims to optimize both efficiency and accuracy, enabling more secure and resource-efficient IoT networks and operating guide. The method approach in this research is multi-purposeful and can be used in all cases of electrical discharge drilling processes with different materials.

REFERENCES

[1]. S. Sareh Ahmadi, Sherif Rashad, Heba Elgazzar, "Efficient feature selection for intrusion detection systems," in *2019 IEEE 10th Annual Ubiquitous Computing, Electronics Mobile Communication Conference (UEMCON)*, 1029-1034, 2019.

[2]. Amir Andalib, Vahid Tabataba Vakili, "A novel dimension reduction scheme for intrusion detection systems in IoT environments," *arXiv:2007.05922*, 2020.

[3]. Taha Archi, Mohammed Benattou, "Efficient dimensionality reduction in intrusion detection systems via weighted power mean for PCA and LDA," in *2024 11th International Conference on Wireless Networks and Mobile Communications (WINCOM)*, 1–6, 2024.

[4]. Raghava Satya SaiKrishna Dittakavi, "Dimensionality reduction based intrusion detection system in cloud computing environment using machine learning," *International Journal of Information and Cybersecurity*, 6, 1, 62–81, 2022.

[5]. Abdulaziz Fatani, Abdelghani Dahou, Mohammed A. A. Al-qaness, Songfeng Lu, Mohamed Abd Abd Elaziz, "Advanced feature extraction and selection approach using deep learning and Aquila optimizer for IoT intrusion detection system," *Sensors*, 22, 1, 140, 2021.

[6]. Mohamed Amine Ferrag, Othmane Friha, Djallel Hamouda, Leandros Maglaras, Helge Janicke, "Edge-IIoTSet: A new comprehensive realistic cyber security dataset of IoT and IIoT applications: Centralized and federated learning," *IEEE Access*, 10, 40281-40306, 2022 2022.

[7]. Michael Greenacre, Patrick J. F. Groenen, Trevor Hastie, Alfonso Iodice D'Enza, Angelos Markos, Elena Tuzhilina, "Principal component analysis," *Nature Reviews Methods Primers*, 2, 1, 2022.

[8]. Firuz Kamalov, Sherif Moussa, Rita Zgheib, Omar Mashaal, "Feature selection for intrusion detection systems," in *2020 13th International Symposium on Computational Intelligence and Design (ISCID)*, 265-269, 2020.

[9]. Sydney M. Kasongo, Yanxia Sun, "Performance analysis of intrusion detection systems using a feature selection method on the UNSW-NB15 dataset," *Journal of Big Data*, 7, 1, 2020.

[10]. Jing Li, Mohd Shahizan Othman, Hewan Chen, Lizawati Mi Yusuf, "Optimizing IoT intrusion detection system: Feature selection versus feature extraction in machine learning," *J. Big Data*, 11, 1, 2024.

[11]. Shancang Li, Li Da Xu, Shanshan Zhao, "The internet of things: A survey," *Information Systems Frontiers*, 17, 2, 243-259, 2014.

[12]. Hung-Jen Liao, Chun-Hung Richard Lin, Ying-Chih Lin, Kuang-Yuan Tung, "Intrusion detection system: A comprehensive review," *Journal of Network and Computer Applications*, 36, 1, 16-24, 2013.

[13]. Nour Moustafa, Jill Slay, "UNSW-NB15: A comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set)," in *2015 Military Communications and Information Systems Conference (MilCIS)*, 1-6, 2015.

[14]. Zhiyuan Tan, Aruna Jamdagni, Xiangjian He, Priyadarsi Nanda, "Network intrusion detection based on LDA for payload feature selection," in *2010 IEEE Globecom Workshops*, 1545-1549, 2010.

[15]. Zhiyuan Tan, Aruna Jamdagni, Xiangjian He, Priyadarsi Nanda, "Network intrusion detection based on LDA for payload feature selection," in *2010 IEEE Globecom Workshops*, 1545-1549, IEEE, 2010.

[16]. Subhash Waskle, Lokesh Parashar, Upendra Singh, "Intrusion detection system using PCA with random forest approach," in *2020 International Conference on Electronics and Sustainable Communication Systems (ICESC)*, 803-808, 2020.

[17]. Karl Pearson, Francis Galton, "VII. note on regression and inheritance in the case of two parents," in *Proceedings of the Royal Society of London*, 58(347-352):240–242, 1895.