IEEE Access
Multidisciplinary : Rapid Review : Open Access Journal

Received 27 September 2024, accepted 12 October 2024, date of publication 16 October 2024, date of current version 11 November 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3481459

RESEARCH ARTICLE

Enhancing Security in Industrial IoT Networks: Machine Learning Solutions for Feature Selection and Reduction

AHMAD HOUKAN[1], ASHWIN KUMAR SAHOO[1], SARADA PRASAD GOCHHAYAT[2], PRABODH KUMAR SAHOO[3], (Member, IEEE), HAIPENG LIU[4], SYED GHUFRAN KHALID[5], AND PRINCE JAIN[3], (Member, IEEE)

[1]Department of Electrical Engineering, C. V. Raman Global University, Bhubaneswar, Odisha 752054, India
[2]Department of Computer Science Engineering, Indian Institute of Technology Jammu, Jammu and Kashmir 181221, India
[3]Department of Mechatronics Engineering, Parul Institute of Technology, Parul University, Vadodara, Gujarat 391760, India
[4]Centre for Intelligent Healthcare, Coventry University, CV1 5RW Coventry, U.K.
[5]School of Science and Technology, Nottingham Trent University, Clifton, NG11 8NF Nottingham, U.K.

Corresponding authors: Syed Ghufran Khalid (syed.khalid@ntu.ac.uk), Prince Jain (princeece48@gmail.com), and Prabodh Kumar Sahoo (sahooprabodhkumar@gmail.com)

This work was supported by the Department of Engineering at Nottingham Trent University.

ABSTRACT The increasing deployment of Internet of Things devices has introduced significant cyber security challenges, creating a need for robust intrusion detection systems. This research focuses on improving anomaly detection in industrial Internet of Things networks through feature reduction and selection. Experiments were performed to compare the effectiveness of Minimum Redundancy Maximum Relevance for feature selection with Principal Component Analysis for feature reduction. Six machine learning algorithms—Decision Trees, k-nearest neighbors, Gaussian Support Vector Machine, Neural Network, Support Vector Machines kernel, and Logistic Regression Kernel—were evaluated for both binary and multi-class classification using feature sets of 4, 12, 23, 50, and 79 features. The results reveal that Minimum Redundancy Maximum Relevance is superior to Principal Component Analysis in identifying crucial features. Notably, Minimum Redundancy Maximum Relevance achieves high accuracy with just 12 features, where the Decision Tree classifier reached an outstanding 99.9% accuracy in binary classification, and k-nearest neighbors achieved 99% accuracy in multi-class classification. The article emphasizes the critical role of feature engineering, with a specific focus on feature selection and reduction, and elaborates on applying MRMR and PCA algorithms to various feature sets. By comparing these methods, it showcases their influence on both model performance and complexity, leading to the development of more efficient and precise intrusion detection systems for Industrial IoT networks. What sets this study apart from previous ones is its novel demonstration of how these techniques significantly reduce training time and model complexity while maintaining or even improving performance, confirming the effectiveness of strategic feature utilization in strengthening Industrial IoT security by balancing accuracy, speed, and model size.

INDEX TERMS Intrusion detection system, Industrial Internet of Things, feature selection, feature reduction, minimum redundancy maximum relevance, principal component analysis.

I. INTRODUCTION

The integration of Internet of Things (IoT) devices into Information Technology (IT) and Operational Technology (OT) domains has resulted in significant technological advancements. However, this integration also introduces considerable cybersecurity challenges that threaten the foundational principles of safety, efficiency, mobility, and security within operational ecosystems. The rise of IoT has created a unique environment where smart devices and cloud services converge, transforming industrial settings [1]. Currently, there are 17 billion connected devices globally, doubling the

The associate editor coordinating the review of this manuscript and approving it for publication was Tai-Hoon Kim.

world's population, making IoT a crucial component of our interconnected society [2], [3], [4]. Cyberattacks are now a pervasive threat, affecting online privacy, social media, businesses, and critical infrastructure [5].

Agile and adaptable strategies are essential for navigating the dynamic IoT ecosystem. IoT continuously transforms the security and risk landscape of interconnected automated systems. The number of cybersecurity threats has surged, from 50 million cyberattacks in 2010 to 900 million in 2019. The annual cost of security breaches is projected to exceed $10.5 trillion US dollars by 2025, highlighting the urgent need for flexible security measures [6]. This research explores the complex field of IIoT networks, where the integration of smart devices and cloud platforms necessitates advanced security measures.

The interconnected nature of IoT devices facilitates the exchange of sensitive data and efficient communication. Cloud-based systems are fundamental to IoT, enabling remote control, data processing, and the application of sophisticated artificial intelligence algorithms [7]. However, this interconnectivity also exposes IIoT devices to various cybersecurity threats, necessitating robust security protocols.

IDS act as vigilant protectors, using a mix of administrative, legal, and technological controls to enhance security, privacy, and confidentiality against unauthorized access [8], [9]. IDSs employing anomaly detection are effective in identifying zero-day attacks, addressing gaps left by traditional signature-based methods [10], [11]. In this context, an intrusion refers to any unexpected activities, that compromise the CIA triad (Confidentiality, Integrity, Availability) [12]. Network traffic, identified by packet header fields, is crucial for anomaly detection. Extracting relevant features from packets helps identify abnormal behaviors indicative of unauthorized usage, reinforcing the CIA principles.

Recent advancements in intrusion detection for IoT environments have integrate of ML methodologies [13]. However, the assumption that IoT devices have uniform feature patterns and packet structures is challenged by the inherent diversity in hardware specifications, functionalities, computational capabilities, and feature generation capacities [14]. Feature dispersion during data aggregation, where attributes often have zero or null values, further complicates the challenge, hindering data modeling accuracy and efficiency. Consequently, feature selection is critical in ML-based intrusion detection solutions to improve detection accuracy and training efficiency. Various techniques, such as Modified Mutual Information Feature Selection (MMIFS) combined with SVM, Flexible Mutual Information Feature Selection (FMIFS), and SVM integrated with NN, have been proposed to enhance the identification of behavioral variables [15], [16].

Despite these efforts, achieving accuracy in anomaly detection remains a significant challenge in the ever-evolving IoT landscape [17]. This paper presents a novel approach to feature selection for ML-based IDS, aiming to improve adaptability and efficiency in the diverse IoT environment. In IIoT networks, characterized by the extensive integration of smart devices and cloud services, the demand for robust security measures is heightened. Traditional defenses, such as firewalls and signature-based IDSs, struggle to cope with the dynamic nature of IIoT, leading to the exploration of innovative methodologies, particularly those leveraging artificial intelligence and ML.

This research utilizes the IOTID20 dataset, capturing complexities in both binary and multi-class classification within IIoT networks. It emphasizes the importance of feature selection algorithms and data processing, focusing on pre-processing and feature extraction. New parameters important to IIoT systems, previously unexplored, were introduced. The study provides an in-depth analysis of feature selection using the IOTID20 dataset and strategic implementation of MRMR (a supervised technique) and PCA (an unsupervised technique). It offers a detailed comparison of their performance, showing that supervised techniques are generally preferred. The feature selection process includes scenarios managing 4, 12, 23, and 50 columns, applied to six ML algorithms: DT, KNN, Gaussian SVM, NN, SVM KERNEL, and Logistic Regression Kernel.

This dual approach skillfully navigates the complex security landscape of industrial IoT, addressing challenges in both binary and multi- class classifications. By applying advanced feature selection and reduction techniques alongside various ML algorithms, the study provides clear insights and solutions for securing advanced industrial IoT networks. The results show superior performance compared to previous studies. Six high-performance ML models are presented for a range of applications, offering a high-performance model with reduced complexity.

The next sections are organized as follows: the Research Goals section defines the aims of this study, the Related Works section surveys previous studies connected to this research, the Methodology section details the methods employed, the Experiment and Results Discussion section covers the experimental design, results, and discussion, and finally, the Conclusion section summarizes the main findings and contributions of the paper.

II. RESEARCH GOALS

A. MAIN OBJECTIVE

The main objectives of this research are as follows:

a) Evaluating the effectiveness of feature selection (MRMR) and feature reduction (PCA) in enhancing anomaly detection within IIoT networks.
b) Reducing computational complexity while maintaining high detection accuracy across multiple machine learning models.
c) Demonstrating the scalability of the proposed dual-feature selection approach in IIoT environments, addressing both binary and multi-class classifications.

B. MAIN CONTRIBUTIONS

Figure 1 illustrates the approach used in our study. This study makes several key contributions:

a) Introduction of a dual feature selection approach combining MRMR and PCA, specifically tailored for anomaly detection in Industrial IoT (IIoT) networks.
b) Comprehensive evaluation of machine learning algorithms for both binary and multi-class classification using the IOTID20 dataset.
c) Demonstrated reduction in computational complexity while maintaining high detection accuracy, making the approach more suitable for real-time IIoT environments.
d) Exploration of new security parameters within the IOTID20 dataset that were previously unexplored, offering valuable insights for improving the performance of intrusion detection systems (IDS).
e) Comparative analysis showing the advantages of MRMR over PCA in maintaining model accuracy and efficiency across various classification tasks.

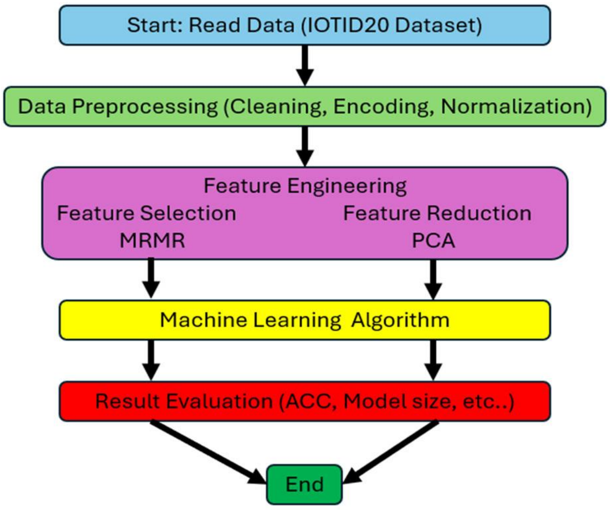

FIGURE 1. Comparative approach flow chart.

III. RELATED WORKS

In the domain of intrusion detection within IIoT networks, identifying intrusions is a crucial function of IDS. This section reviews and analyzes significant works in this field, covering a range of methodologies, algorithms, and datasets. Alkahtani et al. introduced a deep learning-based framework for intrusion detection in IoT networks, using advanced algorithms like CNN, LSTM, and a hybrid CNN-LSTM model. They also employed PSO for feature selection and integrated deep models to enhance anomaly detection efficiency [18].

Indrasiri et al. proposed a real-time detection approach for malicious traffic in both IoT and local networks, emphasizing network-based intrusion detection. Their study combined two publicly available datasets: UNSW-NB15 for local network traffic and IoTID20 for IoT traffic, creating a comprehensive multi-domain dataset with 142,332 records. PCA was used to reduce features to 30 in each dataset for their fusion. Various ML models, including RF and logistic regression, were evaluated. While tree-based models performed well individually, their efficiency decreased when applied to the merged dataset. To address this, the authors developed a stacked ensemble model called Extra Boosting Forest (EBF), combining an extra tree classifier, gradient boosting, and RF. This model achieved 98.5% accuracy for binary classification and 98.4% for multi-class classification, surpassing current state-of-the-art methods. Rigorous statistical tests confirmed the significant performance improvement of EBF over other models [19].

Furqan and Anca Delia Jurcut introduced a novel dataset generation approach, combining traffic data from Software-Defined Networking (SDN), IoT, and traditional IP networks to reflect real-world complexities. The creation of relevant datasets is vital for training robust AI systems. To address class imbalance and overfitting risks, they proposed an innovative Synthetic Data Augmentation Technique (S-DATE), emphasizing smart data augmentation for model generalization. They also developed a Particle Swarm Optimization-based Diverse Self Ensemble Model (PSO-D-SEM) to introduce diversity within the ensemble architecture, highlighting the importance of dataset diversity in improving classification accuracy [20]. This method involves three key stages [21]:

A. INITIAL DATA PROCESSING STAGE

Data preprocessing steps aimed at improving classification outcomes by eliminating duplicate instances, handling missing values, converting non-numeric data to numeric, and standardizing values.

B. DIMENSIONALITY REDUCTION (FEATURE SELECTION) STAGE

This stage is divided into two sub-stages:

a) Feature Ranking using IG and GR Metrics: Features are ranked using Information Gain (IG) and Gain Ratio (GR) filter-based approaches, resulting in top-ranked feature sets for the IoTID20 dataset and the NSL-KDD dataset.
b) Development of Hybrid Feature Selection Method: A hybrid approach uses intersection and union rules to refine feature sets by removing redundant features.

C. MODEL TRAINING AND CLASSIFICATION STAGE

Five ML algorithms (ANN, C4.5, Bagging, KNN, and Ensemble) classify the generated traffic feature sets into normal or intrusive categories for binary classification and multiple categories [22], [23]. Hussein et al. proposed an IDS tailored for IoT embedded environments, using the Meerkat Clan algorithm for feature selection and the RF algorithm for classification. Using the IoTID20 dataset, with identifiable attributes removed, they created a dataset with 79 features in three classes: normal or exploitative activity, type of exploitation, and detailed exploitation descriptions. The optimal number of trees in the RF classifier for both binary and multiclass classification was determined through systematic experimental procedures [24]. Alsulami et al. introduced an intelligent system for intrusion detection and network traffic classification in IoT systems. Using directed ML algorithms such as SNN, DT, BT, SVM, and KNN, the system integrated feature engineering and data preprocessing to improve model performance.

The MRMR algorithm was employed for feature selection, prioritizing essential features. Rigorous evaluation using the IoTID20 dataset, focusing on IoT-specific internet attacks, demonstrated the robustness of intrusion detection and classification models. Feature engineering and data preprocessing were crucial to the high- performance of these models, with the MRMR algorithm playing a key role in identifying and prioritizing features [25].

Ullah et al. conducted a study to enhance intrusion detection within IoT networks using interconnected NN, addressing computational complexities. Using the IoTID20 dataset, their sophisticated model, based on a Deep Convolutional Neural Network (DCNN) optimized with algorithms like Adam and Nadam, showed superior performance across binary-class, multi-class, and subclass detection tasks. The NN outperformed traditional deep learning and ML algorithms, demonstrating reduced computational requirements and efficiency in bolstering IoT network security. Feature selection, facilitated by the Extra Tree Classifier (ETC), meticulously selected 62 impactful features from the original dataset, refining and optimizing the IDS [26]. Sarwar et al. introduced the Improved Dynamic Sticky Binary Particle Swarm Optimization (IDSBPSO) approach for feature selection in anomaly-based IDS for IoT networks. This approach enhances sticky binary PSO by incorporating dynamic search space reduction and parameter modification. Using IoT network datasets (IoTID20 and UNSW-NB15), the IDS employing IDSBPSO for feature selection showed notable performance improvements, often surpassing other PSO techniques in accuracy, even with fewer features. The application of IDSBPSO on resource-constrained IoT devices presents a promising solution, significantly reducing processing costs and prediction timeframes [27]. Bhavsar et al. developed an anomaly-based IDS using a deep learning model called Pearson-Correlation Coefficient Convolutional Neural Network (PCC-CNN) for detecting network anomalies and cyber-attacks within IoT systems. They evaluated three public datasets: NSL-KDD, CICIDS2017, and IOTID20, covering various network attacks like DDoS, scanning, and spoofing. The PCC-CNN model employed Pearson Correlation Co- efficient for feature selection, reducing dimensionality and selecting relevant features, which were then fed into a CNN architecture for classification. Experiments included both binary classification (normal vs. anomaly) and multi-class classification (specific attack types), with comparative evaluation against five traditional ML models. The PCC-CNN model demonstrated high accuracy of 99% for binary classification and 0.91% for multi-class classification, outperforming traditional ML models across various metrics like precision, recall, and F1-score. Additionally, the model efficiently handled imbalanced data and detected anomalies/attacks with limited training samples, exhibiting a low false alarm rate, suitable for real-time intrusion detection [28].

Pawar et al. proposed a model integrating an XGBoost classifier and a modified ANN for IDS using the IoTID20 dataset, comprising network traffic data from smart home devices. Their hybrid deep learning model achieved an accuracy of 90.43%, surpassing other models. Using the Shapiro-Wilk approach for feature ranking, top-ranked features, with over 70% obtaining a ranking higher than 0.50 on the scale, were integrated into cognition models and algorithms, significantly enhancing overall performance [29]. Sarwar et al. [27] aimed to develop an advanced IDS specifically for IoT networks. Their approach used a multimodal methodology encompassing feature selection/reduction, classification, and data preparation techniques. Using RF, XGBoost, and PSO methods, their experiments on the IoTID20 dataset demonstrated 98% accuracy in binary classification and 83% accuracy in multiclass classification [30]. Reflecting on the previous works and methodologies outlined, certain limitations are apparent. These include the use of complex algorithms and models for feature selection without providing comprehensive insights into the significance of each feature, potentially hindering performance across all dataset categories. Additionally, effective IDS design requires well-prepared datasets with appropriate resolution of data-related issues. Reducing high-dimensional datasets through feature selection methods and a deep understanding of these features are crucial for the success of an IDS model. In light of these observations, most studies employ various techniques to select optimal features, including ensemble, filtered, metaheuristic, and unsupervised models. Consequently, in our research, we leveraged strengths and addressed potential weaknesses identified in other studies to achieve superior results.

IV. METHODOLOGY

This study utilizes the IoTID20 dataset [31]. The following outlines the datasets used and details our proposed method for detecting malicious traffic. Figure 2 illustrates our model. Before deploying the IDS in the cloud and applying artificial intelligence algorithms, it is essential to meticulously preprocess the training datasets. Ensuring the integrity of the data, free from errors or missing values, is crucial as it directly affects the accuracy of the outcomes. Given the extensive size and diverse nature of these datasets, which include a wide array of attacks and extraneous information, our initial steps involve configuring and refining the dataset. Subsequently, we identify and select the most relevant features, which form the basis for training the classifier. This classifier will distinguish benign packets from various types of attacks. Thus, our workflow begins with gaining a thorough understanding of the dataset, which is followed by data preparation and feature engineering, and concludes with the training of the classifier.

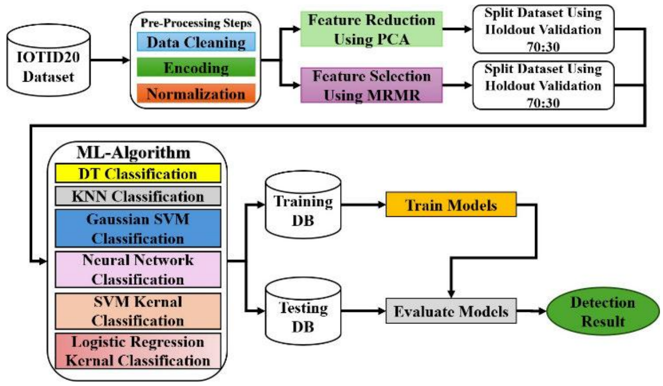

FIGURE 2. Proposed model.

FIGURE 3. IOTID20 dataset testbed environment.

A. OVERVIEW OF THE DATASET IOTID20

The simulated IoT environment used to create the IoTID20 dataset involved an EZVIZ Wi-Fi camera and SKTNGU, a smart home device connected to a Wi-Fi router, reflecting current practices as seen in Figure 3 [25]. Features were extracted from Pcap data using the CIC flowmeter to create related CSV files. The dataset comprises 625,785 records and 86 characteristics, totaling approximately 300MB. The dataset is organized into three categorization categories:

1) Label (Binary categorization): The goal of this categorization style is to distinguish abnormal behavior from typical behavior.
2) Cat (Multi-Classification - 5 sorts of Classification): This category offers a thorough analysis of the dataset by classifying attacks into five different categories.
3) Sub_Cat (Multi-Classification - 9 sorts of Classification): This classification provides a detailed examination of the dataset by further classifying attacks into nine subtypes.

Table 1 provides a comprehensive analysis of the sample distribution across each classification category, highlighting the dataset's composition [31]. Table 2 details the network flow features of the IoTID20 dataset, offering insights into its intricate structure and composition. This dataset is valuable for investigating and understanding security issues in IIoT devices.

TABLE 1. Sample count within the IOTID20 dataset.

Classification type	Classification Statistics	Count
Label (Binary Classification)	Anomaly	585,710
	Normal	40,073
Cat (multi-classification-5type of Classification)	DoS	59,391
	MITM ARP Spoofing	35,377
	Mirai	415,680
	Normal	40,073
	Scan	75,265
Sub_Cat (Multi-Classification-9 type of Classification)	DoS-Synflooding	59,391
	MITM ARP Spoofing	35,377
	Mirai-Ackflooding	55,124
	Mirai-HTTP Flooding	55,818
	Mirai-Hostbruteforceg	121,180
	Mirai-UDP Flooding	183,550
	Normal	40,073
	Scan Hostport	22,192
	Scan Port OS	53,073

B. PRE-PROCESSING STEPS

This phase involves preparing the dataset through three critical steps: Data Cleaning, Encoding, and Normalization.

1) DATA CLEANING

The initial step involves identifying and correcting any inconsistencies, errors, or missing values in the dataset to ensure its accuracy and completeness [32]. Nominal features were initially removed, resulting in a final dataset with 79 features. Columns 21 and 22 (Flow_Byts/s, Flow_Pkts/s) containing infinite values were adjusted by substituting them with the next non-infinite value. Post-cleaning, the column order is as follows: the first column is Src_Port, the second is Dst_Port, the third is Protocol, and the fourth is Flow_Duration, continuing up to column 79 (Idle_Min). Columns 8 to 83 from the original dataset now correspond to columns 4 to 79 after cleaning, with the initial three new columns post-cleaning corresponding to columns 3, 5, and 6 of the original datasets.

2) ENCODING

Following data cleaning, the next step involves encoding, where categorical variables are converted into numerical representations to enhance ML algorithms' interpretability and analysis capabilities [33].

3) NORMALIZATION

Normalization scales the numerical features to a standard range, ensuring uniformity in the data and preventing certain features from dominating others, thus promoting fair and accurate model training [34]. Common normalizing techniques include:

- Z-score normalization (standardization): Scales features to have a mean of zero and a standard deviation of one, useful for features with varying scales or units.

TABLE 2. Network flow features IOTID20 dataset.

Colum_Number	Field	Colum_Number	Field	Colum_Number	Field
1	Flow_ID	30	Fwd_IAT_Max	59	Pkt_Size_Avg
2	Src_IP	31	Fwd_IAT_Min	60	Fwd_Seg_Size_Avg
3	Src_Port	32	Bwd_IAT_Tot	61	Bwd_Seg_Size_Avg
4	Dst_IP	33	Bwd_IAT_Mean	62	Fwd_Byts/b_Avg
5	Dst_Port	34	Bwd_IAT_Std	63	Fwd_Pkts/b_Avg
6	Protocol	35	Bwd_IAT_Max	64	Fwd_Blk_Rate_Avg
7	Timestamp	36	Bwd_IAT_Min	65	Bwd_Byts/b_Avg
8	Flow_Duration	37	Fwd_PSH_Flags	66	Bwd_Pkts/b_Avg
9	Tot_Fwd_Pkts	38	Bwd_PSH_Flags	67	Bwd_Blk_Rate_Avg
10	Tot_Bwd_Pkts	39	Fwd_URG_Flags	68	Subflow_Fwd_Pkts
11	TotLen_Fwd_Pkts	40	Bwd_URG_Flags	69	Subflow_Fwd_Byts
12	TotLen_Bwd_Pkts	41	Fwd_Header_Len	70	Subflow_Bwd_Pkts
13	Fwd_Pkt_Len_Max	42	Bwd_Header_Len	71	Subflow_Bwd_Byts
14	Fwd_Pkt_Len_Min	43	Fwd_Pkts/s	72	Init_Fwd_Win_Byts
15	Fwd_Pkt_Len_Mean	44	Bwd_Pkts/s	73	Init_Bwd_Win_Byts
16	Fwd_Pkt_Len_Std	45	Pkt_Len_Min	74	Fwd_Act_Data_Pkts
17	Bwd_Pkt_Len_Max	46	Pkt_Len_Max	75	Fwd_Seg_Size_Min
18	Bwd_Pkt_Len_Min	47	Pkt_Len_Mean	76	Active_Mean
19	Bwd_Pkt_Len_Mean	48	Pkt_Len_Std	77	Active_Std
20	Bwd_Pkt_Len_Std	49	Pkt_Len_Var	78	Active_Max
21	Flow_Byts/s	50	FIN_Flag_Cnt	79	Active_Min
22	Flow_Pkts/s	51	SYN_Flag_Cnt	80	Idle_Mean
23	Flow_IAT_Mean	52	RST_Flag_Cnt	81	Idle_Std
24	Flow_IAT_Std	53	PSH_Flag_Cnt	82	Idle_Max
25	Flow_IAT_Max	54	ACK_Flag_Cnt	83	Idle_Min
26	Flow_IAT_Min	55	URG_Flag_Cnt	84	Label
27	Fwd_IAT_Tot	56	CWE_Flag_Count	85	Cat
28	Fwd_IAT_Mean	57	ECE_Flag_Cnt	86	Sub_Cat
29	Fwd_IAT_Std	58	Down/Up_Ratio	-	-

- Robust normalization: Less susceptible to outliers, scales features using the interquartile range (IQR).
- Log transformation: Applies logarithms to feature values, particularly useful for handling long-tailed or skewed distributions.
- Min-Max normalization: Scales features within a specified range, typically 0 to 1.
- Min-Max normalization was chosen for its simplicity and effectiveness, ensuring all features influence the analysis and modeling process equally.
- Log transformation: Applies logarithms to feature values, particularly useful for handling long-tailed or skewed distributions.
- Min-Max normalization: Scales features within a specified range, typically 0 to 1.
- Min-Max normalization was chosen for its simplicity and effectiveness, ensuring all features influence the analysis and modeling process equally.

This technique standardizes the scale of input data, enhancing ML algorithms' performance sensitivity to feature magnitude variations.

The formula for Min-Max normalization is:

$$V' = \frac{v - min_A}{max_A - min_A}\,(new\_max_A - new\_min_A) + new\_min_A \tag{1}$$

- V'': This represents the normalized value of the original feature V.
- v : This is the original (raw) value of the feature that you want to normalize.
- min_A : Represents the minimum value of feature A in the dataset.
- max_A : Denotes the maximum value of feature A in the dataset.
- new_min_A : This is the desired minimum value after normalization.
- new_max_A : This is the desired maximum value after normalization

These preprocessing steps create a robust framework for a well-structured and refined dataset, providing a solid foundation for subsequent analysis and modeling.

C. FEATURE ENGINEERING

Feature engineering involves selecting and reducing the number of features in a dataset to enhance model performance and efficiency [35]. Feature selection identifies the most relevant features contributing significantly to the model's predictive ability, thus improving its accuracy and interpretability. A common algorithm for this purpose is MRMR. On the other hand, feature reduction techniques reduce the dataset's dimensionality by transforming or combining features, simplifying the model while preserving essential information. Common methods include PCA for linear dimensionality reduction and T-distributed Stochastic Neighbor Embedding (t-SNE) for nonlinear dimensionality reduction [35].

1) FEATURE REDUCTION

To understand how PCA works, let's explain the fundamental steps and associated equations:

The initial step involves computing the covariance matrix of the dataset.

- Given a data matrix X with dimensions (m, N), where m is the number of samples, and N is the number of features, the covariance matrix is calculated as follows:

$$C = (1/m) * X^T * X \tag{2}$$

Here, X^T is the transpose of matrix X. [36]

Eigenvalues and Eigenvectors: The computation involves determining the eigenvalues and eigenvectors of the covariance matrix C. Each eigenvalue represents the magnitude of variance present in the data, while eigenvectors denote the directions capturing the maximum variance within the dataset.

To calculate the eigenvalues in PCA, the following equation can be utilized:

$$C * v = \lambda * v \tag{3}$$

where:

C is the Covariance Matrix.

λ is the Eigenvalue of the Covariance Matrix.

v is the Eigenvector associated with the Eigenvalue λ.

The process of calculating Eigenvalues is a mathematical operation that relies on solving the above equation. Here's how this process can be done:

a. Iteration Methods: Techniques like Power Iteration or Inverse Iteration can be used to estimate the first Eigenvalue.
b. Start with a Random Vector: A random vector is chosen to start the computation.
c. Refine the Estimate: The estimate for the first Eigenvalue λ_1 and the associated Eigenvector v_1 is refined through iterations. The goal is to obtain an approximate value for the first Eigenvalue λ_1 and the Eigenvector v_1.
d. Estimate Other Eigenvalues: After obtaining the first Eigenvalue, similar processes can be used to calculate other Eigenvalues sequentially.
e. Iteration Continues: The process can be repeated to compute more Eigenvalues and Eigenvectors.

In this way, PCA reduces the dimensionality while retaining the maximum amount of information and variance in the data. The new dimensions are a combination of the original variables, with high eigenvalues gradually decreasing in importance [37]. In this study, the PCA algorithm was applied to the IOTID20 dataset, as shown in Figure 4, highlighting the significance of each column. Columns are arranged in descending order, and our analysis focused on the following cases: 4 columns representing 88.74%, 12 columns representing 99.12%, 23 columns representing 99.9%, and 50 columns representing 99.99% of importance.

2) FEATURE SELECTION

MRMR is an iterative process that aims to balance the trade-off between relevance and redundancy to identify a subset of features that collectively provide valuable information

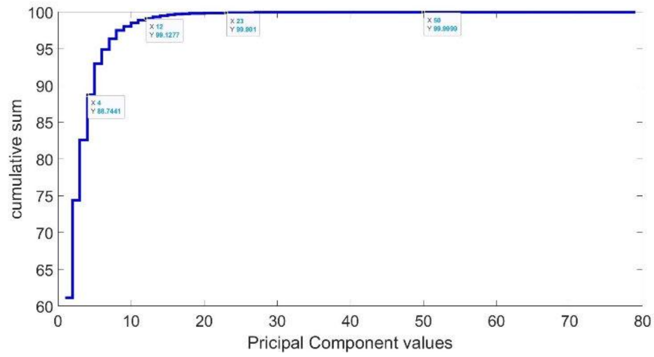

FIGURE 4. **Visualizing feature importance using PCA.**

for predicting the target variable while avoiding information duplication. Widely employed in feature selection, this supervised method enhances the efficiency and interpretability of ML models. MRMR involves a series of steps to calculate relevance and redundancy scores for each feature. The general idea is to maximize the relevance of features to the target variable while minimizing redundancy among the selected features [38], [39].

The MRMR algorithm typically involves the following steps:

1) Relevance Calculation: We compute relevance scores for each feature with respect to the target variable. This could involve calculating mutual information, correlation, or other statistical measures that quantify the relationship between each feature and the target.
   - For each feature X_i, calculate the relevance with the target variable C using mutual information: $Relevance(X_i, C) = I(X_i;C)$
   - Mutual Information $I(X_i;C)$ measures the amount of information that the presence or absence of one feature (X_i) and contributes to the presence or absence of the target variable (C).
2) Redundancy Calculation: We calculate redundancy scores among features. This often involves considering pairwise relationships between features, measuring how much information about one feature is already captured by another.
   - For each pair of features X_i and X_j, calculate the redundancy using mutual information: $Redundancy(X_i, X_j) = I(X_i;X_j)$

Mutual Information, $(I(X_i;X_j))$, measures the amount of information shared between two features (X_i and X_j).

a. Calculation MRMR score:
   For each feature X_i, we calculate the MRMR score using the formula:

$$MRMR\,(X_i) = [I\,(X_i, C) - \frac{1}{|S|}\sum\nolimits_{X_i \in S} I(X_i, X_j)]$$

   This score reflects the balance between the relevance of the feature to the target variable $(I\,(X_i, C))$ and the aver-

TABLE 3. Feature importance scores are sorted using the MRMR algorithm for binary classification.

Number	Feature Column	Feature Scores	Number	Feature Column	Feature Scores	Number	Feature Column	Feature Scores
1	column_2	0.1736	28	column_76	0.0090	54	column_73	0.0047
2	column_53	0.1631	29	column_37	0.0088	55	column_72	0.0047
3	column_12	0.0410	30	column_14	0.0085	56	column_34	0.0036
4	column_30	0.0372	31	column_24	0.0084	57	column_49	0.0036
5	column_4	0.0193	32	column_55	0.0083	58	column_48	0.0032
6	column_1	0.0160	33	column_19	0.0083	59	column_66	0.0029
7	column_45	0.0154	34	column_57	0.0079	60	column_6	0.0029
8	column_50	0.0154	35	column_79	0.0079	61	column_51	0.0027
9	column_69	0.0138	36	column_15	0.0079	62	column_17	0.0022
10	column_77	0.0134	37	column_26	0.0078	63	column_9	0.0021
11	column_10	0.0133	38	column_40	0.0077	64	column_70	0.0017
12	column_78	0.0132	39	column_43	0.0075	65	column_5	0.0016
13	column_44	0.0127	40	column_42	0.0072	66	column_64	0.0016
14	column_65	0.0120	41	column_25	0.0070	67	column_54	0.0013
15	column_28	0.0120	42	column_13	0.0066	68	column_52	0.0012
16	column_67	0.0117	43	column_22	0.0065	69	column_46	3.6140e-04
17	column_23	0.0117	44	column_41	0.0060	70	column_33	0
18	column_21	0.0116	45	column_27	0.0057	71	column_35	0
19	column_56	0.0116	46	column_36	0.0055	72	column_58	0
20	column_3	0.0116	47	column_75	0.0055	73	column_59	0
21	column_8	0.0113	48	column_32	0.0054	74	column_60	0
22	column_18	0.0112	49	column_16	0.0054	75	column_61	0
23	column_11	0.0103	50	column_47	0.0052	76	column_62	0
24	column_7	0.0103	51	column_74	0.0052	77	column_63	0
25	column_38	0.0095	52	column_39	0.0051	78	column_68	0
26	column_31	0.0092	53	column_29	0.0049	79	column_71	0
27	column_20	0.0092						

TABLE 4. Feature importance scores are sorted using the MRMR algorithm for multi-classification.

Number	Feature Column	Feature Scores	Number	Feature Column	Feature Scores	Number	Feature Column	Feature Scores
1	column_1	0.8321	28	column_15	0.3360	54	column_9	0.2289
2	column_47	0.7923	29	column_66	0.3360	55	column_16	0.2190
3	column_4	0.5447	30	column_43	0.3205	56	column_17	0.2176
4	column_69	0.5363	31	column_78	0.3178	57	column_26	0.2176
5	column_12	0.5185	32	column_57	0.3005	58	column_64	0.2141
6	column_52	0.5117	33	column_10	0.2992	59	column_39	0.2141
7	column_2	0.4879	34	column_19	0.2980	60	column_5	0.2130
8	column_6	0.4693	35	column_8	0.2972	61	column_24	0.2092
9	column_31	0.4693	36	column_54	0.2972	62	column_27	0.1961
10	column_75	0.4543	37	column_72	0.2948	63	column_73	0.1720
11	column_50	0.4344	38	column_67	0.2931	64	column_49	0.1520
12	column_3	0.4256	39	column_22	0.2919	65	column_34	0.1520
13	column_32	0.4152	40	column_37	0.2901	66	column_36	0.1365
14	column_77	0.4150	41	column_55	0.2870	67	column_48	0.1365
15	column_14	0.4094	42	column_40	0.2851	68	column_51	0.0968
16	column_53	0.4054	43	column_56	0.2846	69	column_46	0.0714
17	column_21	0.3866	44	column_20	0.2846	70	column_33	0
18	column_38	0.3863	45	column_76	0.2835	71	column_35	0
19	column_42	0.3749	46	column_11	0.2810	72	column_58	0
20	column_28	0.3735	47	column_45	0.2659	73	column_59	0
21	column_41	0.3576	48	column_65	0.2659	74	column_60	0
22	column_29	0.3536	49	column_23	0.2659	75	column_61	0
23	column_74	0.3501	50	column_7	0.2624	76	column_62	0
24	column_70	0.3468	51	column_30	0.2550	77	column_63	0
25	column_79	0.3399	52	column_44	0.2538	78	column_68	0
26	column_13	0.3377	53	column_25	0.2309	79	column_71	0
27	column_18	0.3364						

age redundancy of the feature with the features already selected ($\frac{1}{|S|}\sum_{X_i \in S} I(X_i, X_j)$).
The subtraction term ensures that features with high redundancy with the already selected features are penalized.

b. Feature selection: We select the top-ranked features according to the specified criteria (maximum relevance, minimum redundancy).

c. Iteration (if needed): Depending on the specific MRMR variant, the process might involve iterative steps to refine the selected features.

The MRMR algorithm has been implemented on the IOTID20 dataset, yielding two tables. Table 3 delineates column significance for binary classification, while Table 4 illustrates column significance for multi-class classification, ordered in descending order. Employing a similar analytical

TABLE 5. Comprehensive analysis of binary-classification algorithms: Comparative performance across various feature sets (all features, PCA, MRMR).

Features	Algorithm	ACC (Validation)	AUC	Total Cost (Validation)	Prediction Speed (obs/sec)	Training Time (Sec)	Model Size (Compact)
All Features (79/79)	DT	99.9%	0.9923	263	~190000	296.43	~40 KB
	KNN	99.8%	0.9952	327	~58	11536	~387 KB
	Gaussian SVM	99.5%	0.9848	986	~570	11091	~11 MB
	NN	99.7%	0.9982	543	~94000	4083.9	~33 KB
	SVM Kernel	99.3	0.9833	1251	~19000	53574	~81 KB
	Logistic Regression Kernel	99.3	0.9941	1239	~6300	39845	66 KB
PCA (50/79)	DT	99.2%	0.9481	1495	~79000	290.14	~37KB
	KNN	99.8%	0.9944	364	~97	5891.5	~249 MB
	Gaussian SVM	99.7%	0.995	606	~3400	7541.8	~6 MB
	NN	99.7%	0.9975	475	~100000	2518.2	~23 KB
	SVM Kernel	99.3%	0.9821	1240	~37000	30480	~71 KB
	Logistic Regression Kernel	99.1%	0.9906	1689	~16000	18749	~44 KB
PCA (23/79)	DT	99%	0.9592	1874	~110000	116.78	~33 KB
	KNN	99.8%	0.9954	311	~130	3723	~120 MB
	Gaussian SVM	99.7%	0.9942	573	~12000	4972.6	~3 MB
	NN	99.7%	0.9982	501	~120000	2291.5	~14 KB
	SVM Kernel	99.4%	0.981	1154	~74000	12171	~70 KB
	Logistic Regression Kernel	99.3%	0.9861	1381	~21000	6428.8	~30 KB
PCA (12/79)	DT	99.1%	0.9749	1769	~130000	76.124	~32 KB
	KNN	99.9%	0.9961	275	~360	2073.4	~67 MB
	Gaussian SVM	99.8%	0.9962	448	~28000	2242.7	~1018 KB
	NN	99.6%	0.9976	693	~140000	1808.5	~10 KB
	SVM Kernel	99.3%	0.9676	1286	~44000	1432.1	~14 KB
	Logistic Regression Kernel	98.9%	0.9886	1976	~33000	1769.8	~14 KB
PCA (4/79)	DT	99.1%	0.9859	1717	~110000	55.352	~30 KB
	KNN	99.8%	0.9953	364	~55000	58.781	~40 MB
	Gaussian SVM	95.3%	0.9156	8796	~6800	2254.9	~3 MB
	NN	97.9%	0.9658	3864	~240000	1041.7	~7 KB
	SVM Kernel	98.2%	0.9242	3319	~99000	326.93	~10 KB
	Logistic Regression Kernel	98.2%	0.9556	3466	~95000	156.47	~10 KB
MRMR (50/79)	DT	99.9%	0.9957	249	~260000	201.53	~36 KB
	KNN	99.8%	0.9949	337	~82	5766.9	~249 MB
	Gaussian SVM	99.4%	0.985	1077	~2300	4285.8	~6 MB
	NN	99.7%	0.9983	502	~210000	2635.2	~23 KB
	SVM Kernel	99.4%	0.9853	1144	~39000	29760	~70 KB
	Logistic Regression Kernel	99.3%	0.9928	1392	~20000	14975	~42 KB
MRMR (23/79)	DT	99.9%	0.9986	219	~300000	81.415	~33 KB
	KNN	99.8%	0.9954	316	~220	2860	~120 MB
	Gaussian SVM	99.4%	0.9868	1099	~5100	2214.7	~3 MB
	NN	99.7%	0.9982	559	~270000	2000.4	~14 KB
	SVM Kernel	99.3%	0.9854	1273	~81000	10555	~62 KB
	Logistic Regression Kernel	99.2%	0.9928	1432	~53000	6095.3	~31 KB
MRMR (12/79)	DT	99.9%	0.9987	189	~330000	45.028	~30 KB
	KNN	99.9%	0.9962	281	~330	1831.4	~67 MB
	Gaussian SVM	99.5%	0.9847	1011	~5000	1490.6	~1 MB
	NN	99.7%	0.9974	612	~380000	1694.8	~10 KB
	SVM Kernel	99.3%	0.9701	1381	~38000	807.58	~14 KB
	Logistic Regression Kernel	99.4%	0.9916	1219	~47000	1050.2	~14 KB
MRMR (4/79)	DT	99.1%	0.9814	1701	~270000	26.286	~25 KB
	KNN	99.1%	0.9676	1708	~1300	443.75	~40 MB
	Gaussian SVM	98.4%	0.9151	3031	~13000	1131.5	~1 MB
	NN	98.4%	0.9112	2933	~400000	812.42	~7 KB
	SVM Kernel	98%	0.8515	3808	~120000	177.98	~10 KB
	Logistic Regression Kernel	98.5%	0.9716	2850	~120000	279.69	~10 KB

approach as in PCA, we examined cases with 4 columns, 12 columns, 23 columns, and 50 columns. This comprehensive examination of feature engineering, involving feature selection using MRMR and feature reduction through PCA, reveals that the MRMR algorithm provided profound insights into the dataset by highlighting the significance of each column in both binary and multi-class classification. In contrast, PCA conceals the identity of columns through mathematical transformations and emphasizes their importance in descending order. As a result, our approach to feature reduction distinguishes our study by providing a comprehensive view of the selected features, unlike previous research that did

TABLE 6. Comprehensive analysis of multi-classification algorithms: Comparative performance across various feature sets (all features, PCA, MRMR).

Features	Algorithm	ACC (Validation)	AUC	Total Cost (Validation)	Prediction Speed (obs/sec)	Training Time (Sec)	Model Size (Compact)
All Features (79/79)	DT	97.6%	0.99638	4506	~190000	171.55	~49 KB
	KNN	98.5%	0.99732	2886	~52	10586	~387 MB
	Gaussian SVM	91.8%	0.97946	15384	~170	69248	~120 MB
	NN	91.6%	0.98794	15806	~130000	3485.9	~34 KB
	SVM Kernel	87.8%	0.96944	22836	~1800	1.5962e+05	~685 KB
	Logistic Regression Kernel	90.8%	0.97314	17343	~910	72159	~560 KB
PCA (50/79)	DT	93%	0.97932	13150	~110000	169.26	~46 KB
	KNN	98.4%	0.9972	3002	~87	6478	~249 MB
	Gaussian SVM	93%	0.98986	13152	~230	42604	~66 MB
	NN	93.5%	0.98944	12118	~100000	3117.7	~25 KB
	SVM Kernel	90.9%	0.98086	17126	~3000	53140	~395 KB
	Logistic Regression Kernel	92.2%	0.98238	14596	~1900	44683	~370 KB
PCA (23/79)	DT	93%	0.9793	13148	~140000	74.943	~43 KB
	KNN	98.6%	0.99778	2561	~130	4058.3	~120 MB
	Gaussian SVM	93.4%	0.99006	12327	~1300	15686	~29 MB
	NN	93.4%	0.989	12337	~110000	2442.1	~15 KB
	SVM Kernel	92.4%	0.98344	14235	~8000	16553	~264 KB
	Logistic Regression Kernel	92.3%	0.9831	14506	~3900	7763.8	~214 KB
PCA (12/79)	DT	92.8%	0.97422	13601	~140000	64.82	~41 KB
	KNN	98.9%	0.99834	2118	~250	2117.8	~67 Mb
	Gaussian SVM	96%	0.99226	7551	~2900	5560.5	~13 MB
	NN	92.9%	0.98632	13384	~150000	2228.3	~11 KB
	SVM Kernel	93.4%	0.98316	12453	~13000	5011.4	~143 KB
	Logistic Regression Kernel	93.8%	0.98346	11622	~8800	3414.9	~143 KB
PCA (4/79)	DT	89.7%	0.97224	19276	~98000	60.465	~38 KB
	KNN	98.9%	0.99794	2037	~44000	73.244	~40 Mb
	Gaussian SVM	81.1%	0.91746	35429	~890	5467.1	~16 MB
	NN	84.3%	0.9429	29563	~120000	1401.5	~8 KB
	SVM Kernel	92.5%	0.9629	14037	~40000	1545.1	~100 KB
	Logistic Regression Kernel	92.2%	0.969	14570	~26000	514.15	~100 KB
MRMR (50/79)	DT	97.4%	0.99532	4919	~220000	142.65	~45 KB
	KNN	98.6%	0.99754	2699	~84	6807.7	~249 MB
	Gaussian SVM	91.8%	0.9789	15481	~510	38427	~74 MB
	NN	91.8%	0.98728	15353	~170000	3384.2	~24 KB
	SVM Kernel	91%	0.97674	16954	~3900	97569	~533
	Logistic Regression Kernel	90.5%	0.97366	17797	~1700	34237	~334 KB
MRMR (23/79)	DT	97.6%	0.9965	4514	~330000	43.782	~41 KB
	KNN	98.8%	0.98094	2305	~200	3171.7	~120 MB
	Gaussian SVM	91.8%	0.9965	15359	~1000	13001	~32 MB
	NN	93.2%	0.9877	12845	~260000	2297.7	~15 KB
	SVM Kernel	92.1%	0.9792	14780	~8100	13531	~253 KB
	Logistic Regression Kernel	91.1%	0.97664	16630	~3200	4716.2	~199 KB
MRMR (12/79)	DT	97.6%	0.99666	4508	~390000	28.883	~40 KB
	KNN	99%	0.99786	1887	~310	1829.1	~67 MB
	Gaussian SVM	91.7%	0.98416	15505	~1800	5653.4	~18 MB
	Neural Network	91.5%	0.98656	15900	~300000	2009.8	~11 KB
	SVM Kernel	92.1%	0.98272	14904	~15000	4487.4	~141 KB
	Logistic Regression Kernel	92.1%	0.9823	14817	~12000	3058.4	~142 KB
MRMR (4/79)	DT	94.3%	0.98924	10754	~310000	22.971	~37 KB
	KNN	96.6%	0.99502	6463	~24000	630.54	~40 MB
	Gaussian SVM	88%	0.93168	22559	~1400	4846	~11 MB
	NN	87.8%	0.96288	23128	~470000	1238.5	~8 KB
	SVM Kernel	82.3%	0.92232	33273	~48000	743.9	~99 KB
	Logistic Regression Kernel	85%	0.92896	28234	~39000	628.6	~99 KB

not offer a thorough perspective on features. Additionally, we present practical application examples using the methodology employed in applying ML algorithms.

D. ML ALGORITHM

Machine Learning (ML) relies on several key algorithms, such as Gaussian SVM, KNN, DT, NN, SVM Kernel, and Logistic Regression Kernel, each of which offers distinct advantages for classification tasks. Decision Tree (DT) models are widely used across various domains, including image processing, pattern recognition, and classification [40], [41]. For the DT model, the maximum number of splits was set to 100, with Gini's Diversity Index used as the split criterion, and surrogate decision splits were disabled. KNN, another popular classification method, categorizes data points based on their proximity to existing data points [40]. The algorithm determines the class of a new item by calculating distances

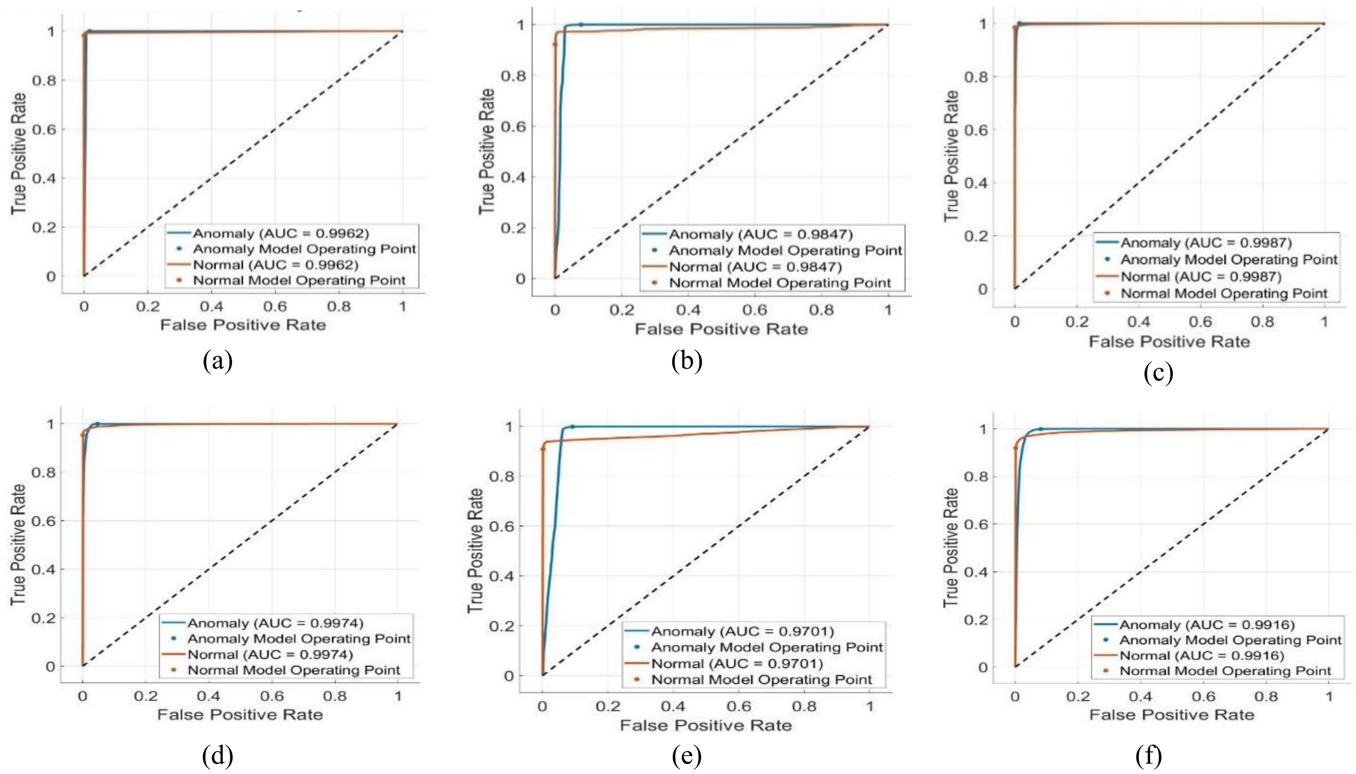

FIGURE 5. ROC curve for MRMR (12/79 features) binary classification. (a) DT; (b) KNN; (c) Gaussian SVM; (d) NN; (e) SVM kernel; (f) Logistic regression kernel.

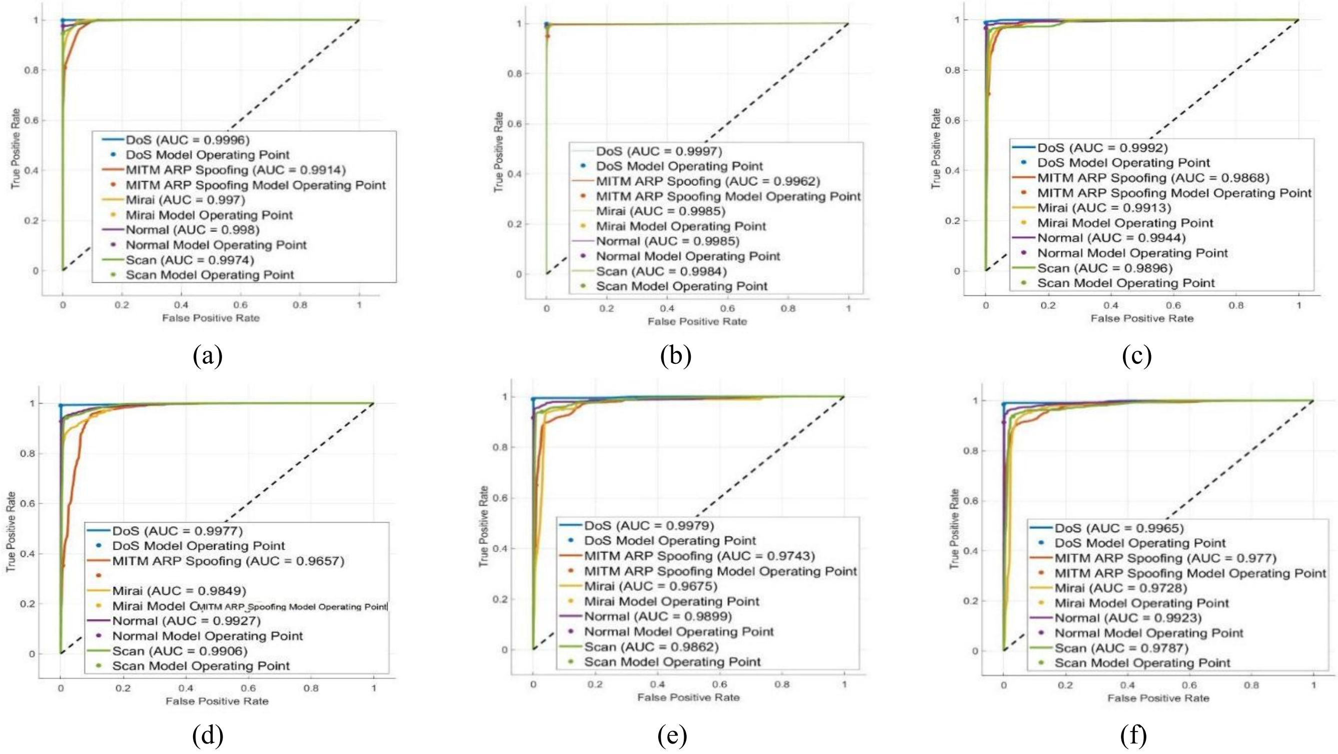

FIGURE 6. ROC curve (12/79 features) binary classification. (a) MRMR-based DT; (b) MRMR-based KNN; (c) PCA-based Gaussian SVM; (d) PCA-based NN; (e) PCA-based SVM kernel; (f) PCA-based logistic regression kernel.

and assigning a class based on the majority of the closest neighbors. For the KNN algorithm, 10 neighbors were used, with the Euclidean distance metric and squared inverse distance weighting. Data was standardized before training [42].

TABLE 7. Multi-modal vs our model-feature engineering and classification metrics comparison table.

Study	Classification	Features Engineer	Number of features selected		Algorithm	ACC		Training Time (Sec)	
			Binary	Multi		Binary	Multi	Binary	Multi
[19]	Binary	Extra Trees Classifier (EXTC)	20		PSO-based diverse-self ensemble model (PSO-D-SEM) with S-DATE	0.989		--	
[23]	Binary, Multi-class	Meerkat Clan Algorithm (MCA)	Not specified		RF	0.999	0.965	340.7	540.46
[25]	Binary, Multi-class	Extra Tree Classifier (ETC)	62		DCNN	0.9984	0.9812	—	
[26]	Binary, Multi-class	IDSBPSO	30		RF	0.998	0.784	266	360
[27]	Binary, Multi-class	Pearson-Correlation Coefficient (PCC)	15	25	PCC-CNN	0.99	0.91	233.17	235.98
[48]	Multi-class	Shapiro-Wilk Technique	12		Hybrid model utilizing modified ANN and XGBoost classifier.	0.9043		-	
[29]	Binary, Multi-class	PSO	17		RF, XGBoost	0.98	0.83		
Proposed Model	Binary	MRMR	12		DT	0.999		45.028	
					KNN	0.999		1831.4	
					Gaussian SVM	0.995		1490.6	
					NN	0.997		1694.8	
					SVM Kernal	0.993		807.58	
					Logistic Regression Kernal	0.994		1050.2	
Proposed Model	Multi-class	MRMR PCA	12		DT	0.976		28.883	
					KNN	0.99		1829.1	
					Gaussian SVM	0.96		5560.5	
					NN	0.929		2228.3	
					SVM Kernal	0.934		5011.4	
					Logistic Regression Kernal	0.938		3414.9	

SVMs are versatile tools for addressing classification, regression, and linear/non-linear problems. Gaussian SVM utilizes hyperplanes to classify data, and the model parameters, including kernel scale and box constraint, were optimized to define the decision boundary that best separates the data into classes [43]. The Gaussian SVM model employed a Gaussian kernel with a kernel scale of 2.2 and a box constraint level of 1. The multiclass coding method was One-Vs-One, and the data was standardized. NN models, inspired by the human brain, are extensively used for tasks requiring the identification of complex patterns, such as image recognition and sentiment analysis. In this NN model, a fully connected architecture with one hidden layer of 25 neurons and the ReLU activation function was implemented. The iteration limit was set to 1000, and no regularization was applied (Lambda = 0). The data was standardized before being input into the network.

SVM with kernel functions enhances the ability to handle non-linear decision boundaries. The kernelized approach transforms the input space through various kernel functions, enabling effective classification in complex feature spaces [44]. For the SVM Kernel model, the learner was set to SVM with automatic selection of expansion dimensions, regularization strength, and kernel scale. The multiclass coding method was One-Vs-One, and the data was standardized. The iteration limit was set to 1000.

Kernel Logistic Regression extends traditional logistic regression by addressing non-linear relationships between features using kernel functions, making it suitable for binary classification tasks [45]. For the Logistic Regression Kernel model, the learner was set to logistic regression with automatic selection of expansion dimensions, regularization strength, and kernel scale. The multiclass coding was One-Vs-One, and the data was standardized, with an iteration limit of 1000.

The classification process involves several steps, including 1D data classification, demonstrating the fitted sigmoidal function and the threshold value. In 2D classification, the boundary obtained for classification in a two-dimensional space is highlighted. In higher-dimensional space, the complex boundary is achieved after mapping to a higher-dimensional feature space. The regularization parameter (λ) plays a critical role in defining the optimal boundary, with different values leading to biased or overfitted cases. Finally, the multi-class classification strategy employs a one-vs-all approach, where each binary logistic regression hypothesis function equals 0.5, and each colored region indicates the respective decision region.

V. EXPERIMENT AND RESULTS DISCUSSION

This section provides a detailed account of the experimental setup, selected evaluation metrics, methodologies used for measurements, and an in-depth discussion of the results obtained from the evaluation of the proposed model.

A. EXPERIMENTAL SETUP

The proposed model's performance assessment was carried out using an ASUS TUF Gaming F15 laptop running Windows 10 Home Single Language. The laptop is equipped with an Intel(R) Core (TM) i5-10300H CPU, operating at 2.50GHz, and it contains 16.0 GB of RAM, with 15.8 GB usable, and GPU NVIDIA GeForce GTX 1650Ti @ 4 GB. For simulation and analysis tasks, MATLAB 2023b was utilized to perform various experiments. This environment provided a robust platform for the creation and evaluation of feature selection and classification algorithms.

B. EVALUATION METRICS

To assess the performance of our proposed model, we compared it against traditional studies using core performance metrics, including accuracy, recall, precision, F-measure, and False Alarm Rate (FAR) [46], [47]. These metrics, crucial for evaluating IDS, are computed based on the confusion matrix:

- True Positive (TP): Instances where an intrusion is accurately identified as an attack.
- True Negative (TN): Instances where normal traffic is correctly identified as normal.
- False Positive (FP): Instances where normal traffic is erroneously classified as an intrusion.
- False Negative (FN): Instances where an intrusion is incorrectly classified as normal traffic.
- ACC: This metric denotes the proportion of correctly identified instances out of the total.

$$Accuracy\,(ACC) = \frac{TP + TN}{TP + TN + FP + FN} \tag{4}$$

- Precision (p): Evaluates the ACC of attack predictions relative to the total predicted attacks.

$$Precision(P) = \frac{TP}{TP + FP} \tag{5}$$

- Recall: This metric delineates the ratio of correctly identified attack instances to the total actual attacks.

$$Recall\,(R) = Detection\ Rate\,(DR) = Sensitivity\,(S) = \text{True Positive Rate (TPR)} = \frac{TP}{TP + FN} \tag{6}$$

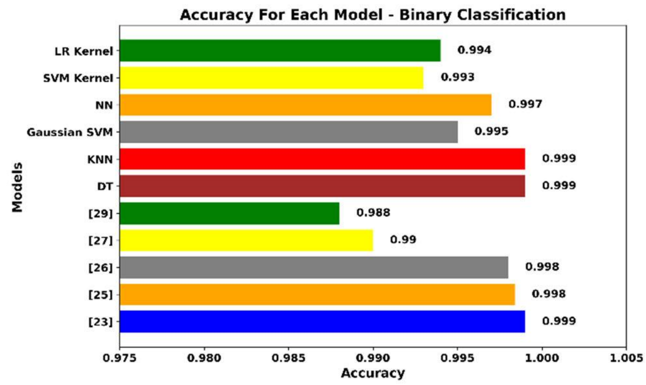

FIGURE 7. Multi-modal vs proposed model-binary classification (ACC).

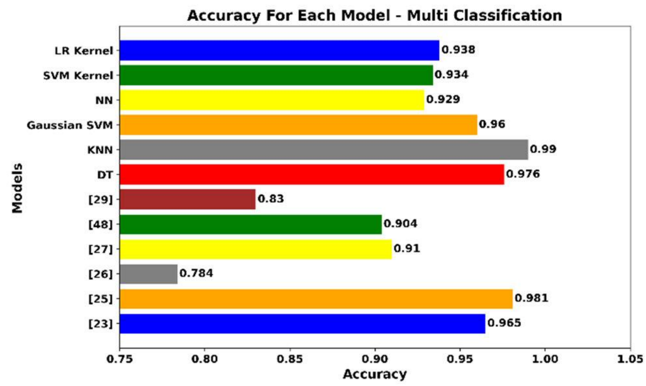

FIGURE 8. Multi-modal vs proposed model-multi classification (ACC).

- F-measure: A composite measure considering both R and precision for system proficiency.

$$F1{-}score = F - measure\,(F) = \frac{2}{\frac{1}{Precision} + \frac{1}{Recall}} \tag{7}$$

- False Alarm Rate: This metric quantifies the percentage of incorrectly predicted attack instances among all actual normal instances.

$$FAR = False\ Positive\ Rate\,(FPR) = \frac{FP}{FP + TN} \tag{8}$$

The ACC, DR, and FAR serve as pivotal metrics for distinguishing between various IDS and assessing their effectiveness. Moreover, we integrate the ROC curve, which compares

TABLE 8. List of abbreviations.

Abbreviation	Full Form
IDS	Intrusion Detection System
IIoT	Industrial Internet of Things
IoT	Internet of Things
ML	Machine Learning
MRMR	Minimum Redundancy Maximum Relevance
PCA	Principal Component Analysis
DT	Decision Trees
KNN	K-Nearest Neighbors
Gaussian SVM	Gaussian Support Vector Machine
SVM	Support Vector Machine
SVM Kernel	Support Vector Machine with Kernel
NN	Neural Network
CNN	Convolutional Neural Networks
LSTM	Long Short-Term Memory
SNN	Shallow Neural Networks
BT	Bagging Trees
ACC	Accuracy
P	Precision
FPR	False Positive Rate
TPR	True Positive Rate
ANN	Artificial Neural Networks
RF	Random Forest
PSO	Particle Swarm Optimization
TP	True Positive
TN	True Negative
FP	False Positive
FN	False Negative
R	Recall
FAR	False Alarm Rate
ROC	Receiver Operating Characteristic

the FPR and TPR of the model. While ROC curves conventionally apply to binary classification models, we extend their utility to the realm of multi-class classification [65,66]. In our research, we assess the effectiveness of our work using several criteria: ACC, Precision (P), R, F1-Score, False FPR, Total Cost (Validation), Prediction Speed (obs/sec), Training Time (Sec), Model Size (Compact), and area under the ROC curve (AUC). We also consider training times. To mitigate overfitting and develop a generalizable IDS, we followed the steps below:

- Split the data into 70% for training and 30% for testing to allow the system to learn patterns from the training set and evaluate its generalization ability using the test set.
- Concurrently applied feature engineering algorithms PCA (for feature reduction) and MRMR (for feature selection). This approach aims to provide a comprehensive understanding of features and the values derived from each configuration, minimizing noise and improving efficiency. From a total of 79 features, sets of 4, 12, 23, and 50 were selected for further analysis, with each feature having 625,785 corresponding records. Subsequently, ML algorithms (DT, KNN, Gaussian SVM, NN, SVM Kernel, and Logistic Regression Kernel) were applied to both PCA and MRMR feature sets.

Next, we compared the best results obtained from each feature set with those from previous studies. Furthermore, we introduced new parameters, such as model size, training time, and ML model hyperparameter, which have not been explicitly addressed in prior research. Consequently, our goal is to provide insights into possible applications based on the obtained results.

Tables 5-6, along with those in APPENDIX B, provide comprehensive analyses, including confusion matrices for both binary and multi-class classification simulations across the chosen ML models. These tables compare the outcomes based on PCA and MRMR applications, each with feature subsets of 4, 12, 23, and 50 features.

C. FINDINGS AND ANALYSIS

The selected model was chosen for its high accuracy (ACC), efficient training time, and compact model size. Compared to previous studies, it performed better with fewer features, particularly in binary classification. Using the MRMR method with 12 features across six ML algorithms (DT, KNN, Gaussian SVM, NN, SVM Kernel, and Logistic Regression Kernel), the DT model achieved the best results, with a training time of 45.028 seconds, a model size of approximately 30 KB, and an ACC of 99.9%.

For multi-class classification, the MRMR method with 12 features also delivered strong results, especially with the DT algorithm, achieving an ACC of 97.6%, a training time of 28.883 seconds, and a model size of around 40 KB. KNN performed well, with an ACC of 99%, though it required a longer training time of 1829.1 seconds, and the model size was 67 KB. The other algorithms (Gaussian SVM, NN, SVM Kernel, and Logistic Regression Kernel) performed better using PCA with 12 features instead of MRMR. Gaussian SVM achieved an ACC of 96% with a training time of 5560.5 seconds, while NN, SVM Kernel, and Logistic Regression Kernel reached accuracies of 92.9%, 93.4%, and 93.8% respectively, with varying training times.

Table 7 compares the proposed models to previous studies in terms of ACC and training time. The DT model stood out in both binary and multi-class classification for its high accuracy, fast training time, and small model size. Although KNN had the highest ACC, it required significantly more time to train. Figures 5 and 6 show the ROC curves, providing insights into the models' performance in both binary and multi-class classifications.

In summary, the use of 12 features—fewer than in previous studies—led to improved or equivalent performance, faster training, smaller model sizes, and lower energy consumption. This is especially important for security systems in IoT environments, where high ACC, quick response times, and efficient energy use are critical. Based on the IDS training results:

The DT model excels in scenarios where high ACC is critical, and computational demands are low, making it ideal for real-time intrusion detection in IoT devices like smart home security or automated industrial controls, where swift threat detection is essential. KNN, on the other hand, suits applications prioritizing ACC over computational efficiency, such as cloud-based security solutions handling large datasets or network traffic analysis to detect sophisticated threats. Gaussian SVM strikes a balance between accuracy and resource use,

TABLE 9. Confusion matrix for each outcome for binary classification and performance metrics for different algorithms.

Features	Algorithm	Confusion Matrix	P	R	F1 - score	FPR
All Features (79/79)	DT	[175697, 16; 247, 11774]	0.999	0.999	0.999	0.00009
	KNN	[175644, 69; 258, 11763]	0.999	0.999	0.999	0.00039
	Gaussian SVM	[175636, 77; 909, 11112]	0.995	0.9241	0.9573	0.00044
	NN	[175640, 73; 470, 11551]	0.9996	0.9973	0.9985	0.0063
	SVM Kernel	[175607, 106; 1160, 10876]	0.9994	0.9934	0.9964	0.0097
	Logistic Regression Kernel	[175634, 79; 1145, 10861]	0.9996	0.9935	0.9965	0.0072
PCA (50/79)	DT	[175646, 67; 1428, 10593]	0.992	0.999	0.996	0.00038
	KNN	[175632, 81; 283, 11738]	0.998	0.999	0.999	0.00046
	Gaussian SVM	[175673, 40; 566, 11455]	0.997	0.999	0.998	0.00023
	NN	[175614, 99; 376, 11645]	0.9994	0.9979	0.9986	0.0084
	SVM Kernel	[175551, 162; 1078, 10943]	0.9991	0.9939	0.9965	0.0146
	Logistic Regression Kernel	[175560, 153; 1536, 10485]	0.9991	0.9913	0.9952	0.0144
PCA (23/79)	DT	[175635, 78; 1796, 10255]	0.99	0.999	0.995	0.00044
	KNN	[175629, 84; 227, 11794]	0.999	0.999	0.999	0.00048
	Gaussian SVM	[175684, 29; 544, 11477]	0.997	0.999	0.998	0.00017
	NN	[175575, 138; 363, 11658]	0.9992	0.9979	0.9986	0.0117
	SVM Kernel	[175501, 212; 942, 11079]	0.9988	0.9947	0.9967	0.0188
	Logistic Regression Kernel	[175612, 101; 1280, 10741]	0.9994	0.9928	0.9961	0.0093
PCA (12/79)	DT	[175464, 249; 1520, 10501]	0.991	0.999	0.995	0.00142
	KNN	[175629, 84; 191, 11830]	0.999	0.999	0.999	0.00048
	Gaussian SVM	[175681, 32; 416, 11605]	0.998	0.999	0.999	0.00018
	NN	[175591, 122; 571, 11450]	0.9895	0.9525	0.9706	0.0007
	SVM Kernel	[175554, 159; 1127, 10894]	0.9856	0.9062	0.9443	0.0009
	Logistic Regression Kernel	[175446, 267; 1709, 10312]	0.9748	0.8578	0.9126	0.0015
PCA (4/79)	DT	[175296, 417; 1300, 10721]	0.993	0.998	0.995	0.00237
	KNN	[175575, 138; 226, 11795]	0.999	0.999	0.999	0.00079
	Gaussian SVM	[174839, 874; 7922, 4099]	0.957	0.995	0.975	0.00497
	NN	[174925, 788; 3076, 8945]	0.9190	0.7441	0.8224	0.0045
	SVM Kernel	[175032, 681; 2638, 9383]	0.9323	0.7806	0.8497	0.0039
	Logistic Regression Kernel	[174771, 942; 2524, 9497]	0.9098	0.79	0.8457	0.0054
MRMR (50/79)	DT	[175698, 15; 234, 11787]	0.999	0.999	0.999	0.00009
	KNN	[175661, 52; 285, 11736]	0.998	0.999	0.999	0.0003
	Gaussian SVM	[175599, 114; 963, 11058]	0.995	0.999	0.997	0.00065
	NN	[175630, 83; 419, 11602]	0.9929	0.9651	0.9788	0.0005
	SVM Kernel	[175575, 138; 1006, 11015]	0.9876	0.9163	0.9506	0.0008
	Logistic Regression Kernel	[175606, 107; 1285, 10736]	0.9901	0.8931	0.9391	0.0006
MRMR (23/79)	DT	[175686, 27; 192, 11829]	0.999	0.999	0.999	0.00015
	KNN	[175657, 56; 260, 11761]	0.999	0.999	0.999	0.00032
	Gaussian SVM	[175605, 108; 991, 11030]	0.995	0.999	0.997	0.00061
	NN	[175624, 89; 470, 11551]	0.9924	0.9609	0.9764	0.0005
	SVM Kernel	[175549, 164; 1109, 10912]	0.9852	0.9077	0.9449	0.0009
	Logistic Regression Kernel	[175511, 202; 1230, 10791]	0.9816	0.8977	0.9378	0.0011
MRMR (12/79)	*DT*	[175703, 10; 179, 11842]	0.999	0.999	0.999	0.00006
	KNN	[175661, 52; 229, 11792]	0.999	0.999	0.999	0.0003
	Gaussian SVM	[175643, 70; 941, 11080]	0.995	0.999	0.997	0.0004
	NN	[175661, 52; 560, 11461]	0.9955	0.9534	0.9740	0.0003
	SVM Kernel	[175432, 281; 1100, 10921]	0.9749	0.9085	0.9405	0.0016
	Logistic Regression Kernel	[175470, 243; 976, 11045]	0.9785	0.9188	0.9477	0.0014
MRMR (4/79)	DT	[175680, 33; 1668, 10353]	0.991	0.999	0.995	0.00019
	KNN	[175625, 88; 1620, 10401]	0.991	0.999	0.995	0.0005
	Gaussian SVM	[175064, 649; 2382, 9639]	0.987	0.996	0.991	0.0037
	NN	[175141, 572; 2361, 9660]	0.9441	0.8036	0.8682	0.0033
	SVM Kernel	[175709, 4; 3804, 8217]	0.9995	0.6836	0.8119	0.00002
	Logistic Regression Kernel	[175262, 451; 2399, 9622]	0.9552	0.8004	0.8710	0.0026

TABLE 10. Confusion matrix for each outcome for multi classification and performance metrics for different algorithms.

Features	Algorithm	Confusion Matrix	P	R	F1 - score	FPR
All Features (79/79)	DT	[17789, 0, 24, 2, 2; 0, 8523, 2021, 54, 15; 0, 744, 123896, 29, 35; 0, 6, 254, 11759, 2; 1, 250, 1053, 14, 21261]	0.999	0.998	0.999	0
	KNN	[17787, 5, 21, 4, 0; 3, 9687, 766, 20, 137; 4, 502, 123729, 23, 446; 1, 53, 194, 11752, 21; 1, 175, 501, 9, 21893]	0.999	0.998	0.999	0.00028
	Gaussian SVM	[17778, 0, 38, 1, 0; 3, 2422, 4520, 40, 3628; 12, 320, 119585, 18, 4769; 8, 13, 499, 11289, 212; 6, 149, 1133, 15, 21276]	0.998	0.999	0.998	0
	NN	[17770, 1, 33, 10, 3; 0, 2020, 8275, 49, 269; 13, 382, 119377, 122, 4810; 10, 5, 468, 11391, 147; 1, 45, 1143, 20, 21370]	0.999	0.997	0.998	0.000056
	SVM Kernel	[17751, 0, 66, 0, 0; 0, 3061, 4128, 8, 3416; 14, 773, 114054, 60, 9803; 6, 21, 760, 9991, 1243; 4, 140, 2352, 42, 20041]	0.999	0.996	0.997	0
	Logistic Regression Kernel	[17727, 0, 89, 1, 0; 0, 3401, 3647, 1, 3564; 15, 986, 118573, 113, 5017; 10, 13, 812, 9697, 1507; 7, 184, 1323, 54, 21011]	0.998	0.995	0.996	0
PCA (50/79)	DT	[17610, 25, 119, 16, 47; 0, 6983, 3217, 42, 371; 4, 2618, 119423, 106, 2553; 6, 47, 1349, 10207, 412; 1, 454, 1746, 17, 20361]	0.999	0.988	0.994	0.00142
	KNN	[17779, 5, 25, 8, 0; 1, 9644, 790, 25, 153; 3, 526, 123681, 24, 470; 2, 56, 222, 11719, 22; 2, 168, 487, 13, 21909]	0.999	0.998	0.999	0.00028
	Gaussian SVM	[17358, 3, 456, 0, 0; 0, 5066, 5343, 8, 196; 1, 1821, 119236, 10, 3636; 0, 47, 513, 11454, 7; 0, 251, 852, 8, 21468]	0.999	0.974	0.987	0.00017
	NN	[17793, 1, 17, 6, 0; 1, 3906, 6566, 43, 97; 7, 1793, 121085, 140, 1679; 7, 24, 361, 11540, 89; 1, 178, 1069, 39, 21292]	0.999	0.999	0.999	0.000056
	SVM Kernel	[17562, 0, 249, 6, 0; 2, 1818, 7929, 49, 815; 19, 178, 119076, 111, 5320; 18, 6, 658, 11105, 234; 7, 22, 1483, 20, 21047]	0.997	0.986	0.991	0
	Logistic Regression Kernel	[17372, 0, 440, 5, 0; 1, 4868, 3357, 31, 2356; 18, 1048, 118816, 117, 4705; 4, 17, 857, 10884, 259; 5, 193, 1159, 24, 21198]	0.998	0.975	0.986	0
PCA (23/79)	DT	[17615, 24, 119, 11, 48; 0, 6946, 3294, 10, 363; 1, 2617, 119503, 53, 2530; 1, 50, 1443, 10113, 414; 2, 448, 1709, 11, 20409]	0.999	0.989	0.994	0.00136
	KNN	[17770, 5, 35, 6, 1; 2, 9798, 653, 23, 137; 3, 421, 123849, 24, 407; 5, 54, 156, 11785, 21; 1, 152, 436, 19, 21971]	0.999	0.997	0.998	0.00028
	Gaussian SVM	[17476, 7, 332, 2, 0; 1, 5167, 5390, 6, 49; 0, 1857, 119801, 6, 3040; 1, 60, 486, 11469, 5; 0, 235, 842, 8, 21494]	0.999	0.981	0.99	0.0004
	NN	[17747, 9, 49, 10, 2; 1, 3856, 6548, 59, 149; 0, 1758, 121162, 168, 1616; 4, 32, 470, 11461, 54; 0, 157, 1213, 38, 21171]	0.999	0.996	0.999	0.0005
	SVM Kernel	[17582, 2, 227, 6, 0; 1, 4800, 5575, 21, 216; 11, 1260, 118589, 90, 4754; 14, 38, 654, 11298, 17; 4, 196, 1128, 21, 21230]	0.998	0.987	0.992	0.00011
	Logistic Regression Kernel	[17451, 2, 360, 3, 1; 0, 4929, 3348, 51, 2285; 2, 1079, 118714, 112, 4797; 7, 34, 839, 10949, 192; 3, 192, 1167, 32, 21185]	0.999	0.979	0.989	0.00011
PCA (12/79)	DT	[17499, 5, 198, 37, 78; 11, 6500, 3438, 20, 644; 7, 2031, 118866, 148, 3652; 6, 40, 848, 10104, 923; 3, 163, 1222, 27, 21164]	0.998	0.982	0.99	0.00028
	KNN	[177751, 7, 45, 13, 1; 2, 9919, 563, 21, 108; 7, 346, 124039, 24, 288; 5, 54, 121, 11822, 19; 2, 121, 346, 25, 22085]	0.999	0.999	0.999	0.00004
	Gaussian SVM	[17596, 24, 195, 2, 0; 0, 7476, 3088, 22, 27; 1, 883, 121946, 11, 1863; 0, 89, 313, 11608, 11; 0, 243, 772, 7, 21557]	0.999	0.988	0.994	0.00136
	NN	[17644, 25, 113, 27, 8; 3, 3734, 6546, 80, 250; 27, 941, 120572, 281, 2883; 5, 53, 799, 11146, 18; 8, 85, 1184, 48, 21254]	0.997	0.99	0.994	0.00141
	SVM Kernel	[17616, 23, 170, 7, 1; 0, 6892, 3698, 11, 12; 11, 1349, 118558, 90, 4696; 1, 56, 948, 11001, 15; 3, 188, 1140, 34, 21214]	0.999	0.989	0.994	0.0013
	Logistic Regression Kernel	[17541, 23, 238, 10, 5; 0, 7477, 2941, 16, 179; 3, 873, 118976, 60, 4792; 0, 52, 961, 10969, 39; 2, 182, 1222, 24, 21149]	0.999	0.984	0.992	0.0013

making it effective for high-stakes environments like financial transaction monitoring or smart grid security, where both precision and efficiency are key. NN models are useful for recognizing complex patterns and are suitable for advanced malware detection or behavior analysis in enterprise networks. SVM Kernel, offering good accuracy with moderate resource requirements, is commonly applied in healthcare IoT security to protect medical devices or wearable health monitors from data breaches. Lastly, Logistic Regression Kernel works well for simpler classification tasks with low to moderate computational demands, such as phishing detection in emails or spam filtering, where fast, reliable classification of benign versus malicious content is necessary without using heavy resources.

These methods present diverse options for different security challenges, ensuring optimal performance while managing resource constraints. Figures 7 and 8 illustrate the enhanced performance of this method compared to previous studies, showing either higher or comparable accuracy in both binary and multi-class classifications. This was achieved with fewer features, resulting in models that are not only more accurate but also smaller in size and quicker in response time.

VI. CONCLUSION

The limited computational resources in IoT systems and networks can make it difficult to train, validate, and implement attack classification models for cybersecurity. To address these challenges, reducing the number of features is important

TABLE 10. *(Continued.)* Confusion matrix for each outcome for multi classification and performance metrics for different algorithms.

PCA (4/79)	DT	[17367, 18, 330, 102, 0; 0, 3936, 6538, 96, 43; 0, 146, 121322, 1878, 1358; 0, 33, 2877, 9093, 18; 2, 51, 4192, 1594, 16740]	0.999	0.975	0.987	0.00104
	KNN	[17712, 3, 78, 19, 5; 5, 9970, 509, 28, 101; 6, 309, 124102, 45, 242; 7, 56, 145, 11798, 15; 3, 113, 315, 33, 22115]	0.999	0.975	0.987	0.00104
	Gaussian SVM	[17501, 27, 247, 26, 16; 1, 1714, 6403, 52, 2443; 45, 197, 116086, 1335, 7041; 1, 18, 5244, 6041, 717; 11, 13, 9983, 1609, 10963]	0.997	0.982	0.989	0.00154
	NN	[17438, 30, 240, 32, 77; 0, 1755, 6178, 81, 2599; 31, 213, 119285, 1550, 3625; 29, 31, 2206, 8970, 785; 6, 16, 8553, 3281, 10723]	0.996	0.979	0.987	0.00171
	SVM Kernel	[17360, 41, 392, 9, 15; 0, 6365, 4102, 12, 134; 4, 735, 119003, 143, 4819; 0, 61, 1719, 10104, 137; 0, 56, 1552, 106, 20865]	0.999	0.974	0.987	0.00235
	Logistic Regression Kernel	[17349, 30, 393, 35, 10; 0, 6424, 3940, 32, 217; 0, 810, 118998, 184, 4712; 1, 51, 2121, 9645, 203; 3, 118, 1616, 94, 20748]	0.999	0.974	0.986	0.00172
MRMR (50/79)	DT	[17778, 2, 30, 7, 0; 0, 8576, 1950, 75, 13; 0, 813, 123803, 46, 41; 1, 16, 279, 11725, 0; 1, 290, 1332, 23, 20933]	0.999	0.998	0.999	0.00011
	KNN	[17781, 2, 23, 9, 2; 2, 9764, 700, 28, 120; 2, 476, 123778, 31, 416; 0, 56, 178, 11775, 12; 0, 171, 461, 10, 21937]	0.999	0.998	0.999	0.00011
	Gaussian SVM	[17766, 1, 48, 2, 0; 0, 1979, 4931, 36, 3668; 8, 118, 119849, 24, 4704; 6, 11, 449, 11331, 224; 0, 84, 1146, 21, 21328]	0.999	0.997	0.998	0.00006
	NN	[17765, 3, 30, 13, 6; 1, 2590, 7042, 18, 963; 6, 968, 119349, 166, 4214; 11, 15, 429, 11291, 275; 0, 128, 1040, 25, 21386]	0.999	0.997	0.998	0.00016
	SVM Kernel	[17754, 1, 56, 6, 0; 0, 2884, 4087, 12, 3631; 6, 602, 118575, 100, 5420; 0, 12, 559, 10412, 1038; 0, 114, 1278, 32, 21155]	0.999	0.996	0.998	0.000056
	Logistic Regression Kernel	[17743, 1, 67, 6, 0; 0, 3022, 4079, 4, 3509; 8, 847, 118689, 105, 5054; 3, 6, 818, 9736, 1458; 2, 134, 1621, 75, 20747]	0.999	0.996	0.997	0. 000056
MRMR (23/79)	DT	[17783, 2, 19, 13, 0; 0, 8388, 2136, 75, 15; 0, 582, 124045, 46, 30; 3, 19, 279, 11719, 1; 0, 272, 1000, 22, 21285]	0.999	0.998	0.999	0.00011
	KNN	[17779, 2, 22, 13, 1; 2, 9900, 573, 25, 114; 2, 382, 123916, 29, 374; 3, 55, 151, 11805, 7; 1, 151, 391, 7, 22029]	0.999	0.998	0.999	0.00011
	Gaussian SVM	[17762, 2, 42, 9, 2; 0, 2220, 4704, 15, 3675; 2, 330, 119626, 24, 4721; 2, 6, 418, 11374, 221; 0, 77, 1094, 15, 21393]	0.999	0.997	0.999	0.00011
	NN	[17771, 1, 37, 8, 0; 0, 3324, 7152, 43, 95; 0, 1662, 121132, 155, 1754; 10, 26, 469, 11299, 217; 0, 164, 1025, 27, 21363]	0.999	0.997	0.998	0.000056
	SVM Kernel	[17750, 2, 48, 15, 2; 0, 3457, 4275, 12, 2870; 8, 626, 119278, 74, 4717; 0, 14, 491, 11227, 289; 0, 127, 1172, 38, 21242]	0.999	0.996	0.998	0.00011
	Logistic Regression Kernel	[17716, 0, 81, 20, 0; 0, 3036, 3938, 5, 3635; 7, 889, 119099, 90, 4618; 3, 12, 726, 10194, 1086; 2, 178, 1286, 54, 21059]	0.999	0.994	0.997	0
MRMR (12/79)	DT	[17787, 0, 17, 13, 0; 0, 8588, 1932, 78, 16; 1, 855, 123768, 50, 29; 3, 9, 279, 11730, 0; 0, 295, 909, 22, 21353]	0.999	0.998	0.999	0
	KNN	[17779, 3, 18, 15, 2; 2, 10071, 438, 14, 89; 2, 404, 123994, 29, 274; 5, 49, 110, 11855, 2; 0, 128, 296, 7, 22148]	0.999	0.998	0.999	0.00017
	Gaussian SVM	[17754, 1, 49, 11, 2; 1, 2762, 4060, 11, 3780; 4, 765, 118953, 87, 4894; 3, 9, 378, 11490, 141; 5, 147, 1122, 8, 21270]	0.999	0.996	0.998	0.00006
	NN	[17769, 1, 36, 9, 2; 0, 2654, 7838, 31, 91; 2, 916, 118960, 408, 4417; 11, 24, 745, 11148, 93; 0, 172, 1080, 24, 21303]	0.999	0.997	0.998	0.000056
	SVM Kernel	[17729, 0, 67, 19, 2; 0, 3513, 6010, 14, 1077; 1, 652, 118880, 256, 4914; 2, 6, 459, 11463, 91; 1, 90, 1232, 11, 21245]	0.999	0.995	0.997	0
	Logistic Regression Kernel	[17701, 2, 85, 26, 3; 0, 4365, 4364, 11, 1874; 4, 785, 118663, 217, 5034; 6, 17, 679, 11109, 210; 2, 181, 1304, 13, 21079]	0.999	0.993	0.996	0.00011
MRMR (4/79)	DT	[17771, 1, 30, 14, 1; 0, 6550, 3985, 41, 38; 0, 359, 121272, 51, 3021; 5, 16, 435, 10200, 1365; 1, 181, 1160, 50, 21187]	0.999	0.997	0.999	0.00006
	KNN	[17774, 4, 26, 12, 1; 1, 9962, 497, 45, 109; 1, 404, 122312, 305, 1681; 4, 49, 576, 10832, 560; 1, 127, 1763, 297, 20391]	0.999	0.998	0.999	0.00023
	Gaussian SVM	[17756, 0, 43, 18, 0; 0, 1710, 4940, 40, 3924; 5, 258, 119363, 128, 4949; 3, 1, 2055, 9772, 190; 0, 7, 5989, 9, 16574]	0.999	0.997	0.998	0
	NN	[17762, 3, 32, 12, 8; 0, 1980, 5254, 22, 3358; 0, 558, 117961, 372, 5812; 17, 10, 1977, 9779, 238; 0, 177, 5233, 45, 17124]	0.999	0.997	0.998	0.00016
	SVM Kernel	[17706, 1, 109, 1, 0; 0, 57, 9442, 0, 1115; 3, 22, 119412, 3, 5263; 2, 1, 11818, 0, 200; 0, 8, 5285, 0, 17286]	0.999	0.994	0.997	0.000056
	Logistic Regression Kernel	[17610, 0, 194, 10, 3; 7, 1557, 8224, 90, 736; 29, 87, 120281, 565, 3741; 4, 7, 5475, 6524, 11; 2, 3, 9018, 28, 13528]	0.998	0.988	0.993	0

to create lightweight and efficient models. This study focused on reducing the number of features by evaluating feature selection and extraction techniques. A detailed evaluation of feature selection methods was conducted, setting this work apart from previous research by emphasizing data preparation and classifier training. The analysis showed that selecting 12 out of 79 features accounted for 99% of their significance. This selection, combined with six ML algorithms, produced results equal to or better than prior studies. In binary classification, the DT algorithm achieved 99.9% accuracy with a training time of 45.028 seconds. KNN also reached 99.9% accuracy with a model size of 30 KB. Other algorithms, such as Gaussian SVM, NN, SVM Kernel, and Logistic Regression Kernel, achieved accuracy rates between 99.3% and 99.7%. For multi-class classification using the MRMR method, DT achieved 97.6% accuracy, KNN reached 99%, and other algorithms showed accuracy between 92.9% and 96%. This study successfully reduced model complexity while improving accuracy using MRMR and PCA techniques. It demonstrated that these methods are scalable and flexible for industrial IIoT applications by identifying important features with minimal computational effort. However, there were limitations, such as longer training times for KNN and Gaussian SVM. Additionally, parameters like model size and prediction speed were evaluated, but direct comparisons to other studies were difficult. Applying these findings to more complex datasets or real-world scenarios may present challenges, and future work will focus on using deep learning techniques and testing with the latest datasets to further improve performance.

APPENDIX A

Table 8 lists the abbreviations employed in the manuscript, which have not been expanded upon within the document. These abbreviations are presented in the sequence of their initial occurrence.

APPENDIX B

Tables 9 and 10 display the confusion matrices and performance metrics for various ML algorithms applied in both binary and multi-class classification tasks.

VII. ACKNOWLEDGMENT

The author(s) would like to express sincere gratitude to the Department of Engineering at Nottingham Trent University for their invaluable support and resources that made this research possible.

REFERENCES

[1] A. B. Kathole, K. N. Vhatkar, A. Goyal, S. Kaushik, A. S. Mirge, P. Jain, M. S. Soliman, and M. T. Islam, "Secure federated cloud storage protection strategy using hybrid heuristic attribute-based encryption with permissioned blockchain," *IEEE Access*, vol. 12, pp. 117154–117169, 2024, doi: 10.1109/access.2024.3447829.

[2] H. Alkahtani, T. H. H. Aldhyani, and M. Al-Yaari, "Adaptive anomaly detection framework model objects in cyberspace," *Appl. Bionics Biomechanics*, vol. 2020, pp. 1–14, Dec. 2020, doi: 10.1155/2020/6660489.

[3] M. Tang, M. Alazab, and Y. Luo, "Big data for cybersecurity: Vulnerability disclosure trends and dependencies," *IEEE Trans. Big Data*, vol. 5, no. 3, pp. 317–329, Sep. 2019, doi: 10.1109/TBDATA.2017.2723570.

[4] H. Nizam, S. Zafar, Z. Lv, F. Wang, and X. Hu, "Real-time deep anomaly detection framework for multivariate time-series data in industrial IoT," *IEEE Sensors J.*, vol. 22, no. 23, pp. 22836–22849, Dec. 2022, doi: 10.1109/JSEN.2022.3211874.

[5] A. A. Smadi, B. T. Ajao, B. K. Johnson, H. Lei, Y. Chakhchoukh, and Q. Abu Al-Haija, "A comprehensive survey on cyber-physical smart grid testbed architectures: Requirements and challenges," *Electronics*, vol. 10, no. 9, p. 1043, Apr. 2021, doi: 10.3390/electronics10091043.

[6] J. Fox. *Top Cybersecurity Statistics for 2024*. Accessed: Jul. 2, 2024. [Online]. Available: https://www.cobalt.io/blog/cybersecurity-statistics-2024

[7] K. Agrawal and N. Kamboj, "Smart agriculture using IoT: A futuristic approach," *Int. J. Inf. Dissemination Technol.*, vol. 9, no. 4, p. 186, 2019, doi: 10.5958/2249-5576.2019.00036.0.

[8] T. H. H. Aldhyani, M. Alrasheedi, A. A. Alqarni, M. Y. Alzahrani, and A. M. Bamhdi, "Intelligent hybrid model to enhance time series models for predicting network traffic," *IEEE Access*, vol. 8, pp. 130431–130451, 2020, doi: 10.1109/ACCESS.2020.3009169.

[9] P. Sunhare, R. R. Chowdhary, and M. K. Chattopadhyay, "Internet of Things and data mining: An application oriented survey," *J. King Saud Univ. Comput. Inf. Sci.*, vol. 34, no. 6, pp. 3569–3590, Jun. 2022, doi: 10.1016/j.jksuci.2020.07.002.

[10] J. Li, M. S. Othman, H. Chen, and L. M. Yusuf, "Optimizing IoT intrusion detection system: Feature selection versus feature extraction in machine learning," *J. Big Data*, vol. 11, no. 1, pp. 1–44, Feb. 2024, doi: 10.1186/s40537-024-00892-y.

[11] M. Eskandari, Z. H. Janjua, M. Vecchio, and F. Antonelli, "Passban IDS: An intelligent anomaly-based intrusion detection system for IoT edge devices," *IEEE Internet Things J.*, vol. 7, no. 8, pp. 6882–6897, Aug. 2020, doi: 10.1109/JIOT.2020.2970501.

[12] A. Abraham, C. Grosan, and C. Martin-Vide, "Evolutionary design of intrusion detection programs," *Int. J. Netw. Secur.*, vol. 4, no. 3, pp. 328–339, 2007.

[13] A. Verma and V. Ranga, "Machine learning based intrusion detection systems for IoT applications," *Wireless Pers. Commun.*, vol. 111, no. 4, pp. 2287–2310, Apr. 2020, doi: 10.1007/s11277-019-06986-8.

[14] H. Alrubayyi, G. Goteng, M. Jaber, and J. Kelly, "Challenges of malware detection in the IoT and a review of artificial immune system approaches," *J. Sensor Actuator Netw.*, vol. 10, no. 4, p. 61, Oct. 2021, doi: 10.3390/jsan10040061.

[15] M. A. Ambusaidi, X. He, P. Nanda, and Z. Tan, "Building an intrusion detection system using a filter-based feature selection algorithm," *IEEE Trans. Comput.*, vol. 65, no. 10, pp. 2986–2998, Oct. 2016, doi: 10.1109/TC.2016.2519914.

[16] A. H. Sung and S. Mukkamala, "Identifying important features for intrusion detection using support vector machines and neural networks," in *Proc. Symp. Appl. Internet*, 2003, pp. 209–216, doi: 10.1109/saint.2003.1183050.

[17] A. Khraisat, I. Gondal, P. Vamplew, and J. Kamruzzaman, "Survey of intrusion detection systems: Techniques, datasets and challenges," *Cybersecurity*, vol. 2, no. 1, pp. 1–22, Dec. 2019, doi: 10.1186/s42400-019-0038-7.

[18] H. Alkahtani and T. H. H. Aldhyani, "Intrusion detection system to advance Internet of Things infrastructure-based deep learning algorithms," *Complexity*, vol. 2021, no. 1, pp. 1–18, Jan. 2021, doi: 10.1155/2021/5579851.

[19] O. O. Petinrin, F. Saeed, X. Li, F. Ghabban, and K.-C. Wong, "Malicious traffic detection in IoT and local networks using stacked ensemble classifier," *Comput., Mater. Continua*, vol. 71, no. 1, pp. 489–515, 2022, doi: 10.32604/cmc.2022.019636.

[20] F. Rustam and A. D. Jurcut, "Malicious traffic detection in multi-environment networks using novel S-DATE and PSO-D-SEM approaches," *Comput. Secur.*, vol. 136, Jan. 2024, Art. no. 103564, doi: 10.1016/j.cose.2023.103564.

[21] U. Chauhan, H. Chhabra, P. Jain, A. Dev, N. Chauhan, and B. Kumar, "Chaos inspired invasive weed optimization algorithm for parameter estimation of solar PV models," *IFAC J. Syst. Control*, vol. 27, Mar. 2024, Art. no. 100239, doi: 10.1016/j.ifacsc.2023.100239.

[22] P. K. Sahoo, M. K. Panda, U. Panigrahi, G. Panda, P. Jain, M. S. Islam, and M. T. Islam, "An improved VGG-19 network induced enhanced feature pooling for precise moving object detection in complex video scenes," *IEEE Access*, vol. 12, pp. 45847–45864, 2024, doi: 10.1109/ACCESS.2024.3381612.

[23] K. Albulayhi, Q. Abu Al-Haija, S. A. Alsuhibany, A. A. Jillepalli, M. Ashrafuzzaman, and F. T. Sheldon, "IoT intrusion detection using machine learning with a novel high performing feature selection method," *Appl. Sci.*, vol. 12, no. 10, p. 5015, May 2022, doi: 10.3390/app12105015.

[24] B. Shingala, P. Panchal, S. Thakor, P. Jain, A. Joshi, C. R. Vaja, R. K. Siddharth, and V. A. Rana, "Random forest regression analysis for estimating dielectric properties in epoxy composites doped with hybrid nano fillers," *J. Macromolecular Sci., B*, vol. 63, no. 12, pp. 1297–1311, Dec. 2024, doi: 10.1080/00222348.2024.2322189.

[25] A. A. Alsulami, Q. Abu Al-Haija, A. Tayeb, and A. Alqahtani, "An intrusion detection and classification system for IoT traffic with improved data engineering," *Appl. Sci.*, vol. 12, no. 23, p. 12336, Dec. 2022, doi: 10.3390/app122312336.

[26] S. Ullah, J. Ahmad, M. A. Khan, E. H. Alkhammash, M. Hadjouni, Y. Y. Ghadi, F. Saeed, and N. Pitropakis, "A new intrusion detection system for the Internet of Things via deep convolutional neural network and feature engineering," *Sensors*, vol. 22, no. 10, p. 3607, May 2022, doi: 10.3390/s22103607.
[27] A. Sarwar, A. M. Alnajim, S. N. K. Marwat, S. Ahmed, S. Alyahya, and W. U. Khan, "Enhanced anomaly detection system for IoT based on improved dynamic SBPSO," *Sensors*, vol. 22, no. 13, p. 4926, Jun. 2022, doi: 10.3390/s22134926.
[28] M. A. Alsoufi, S. Razak, M. M. Siraj, I. Nafea, F. A. Ghaleb, F. Saeed, and M. Nasser, "Anomaly-based intrusion detection systems in IoT using deep learning: A systematic literature review," *Appl. Sci.*, vol. 11, no. 18, p. 8383, Sep. 2021, doi: 10.3390/app11188383.
[29] G. Pawar and J. Agarkhed, "Efficient trust inference model for pervasive computing based on hybrid deep learning," *Int. J. Intell. Syst. Appl. Eng.*, vol. 11, no. 2, pp. 170–179, 2023.
[30] A. Sarwar, S. Hasan, W. U. Khan, S. Ahmed, and S. N. K. Marwat, "Design of an advance intrusion detection system for IoT networks," in *Proc. 2nd Int. Conf. Artif. Intell. (ICAI)*, Mar. 2022, pp. 46–51, doi: 10.1109/ICAI55435.2022.9773747.
[31] I. Ullah and Q. H. Mahmoud, "A scheme for generating a dataset for anomalous activity detection in IoT networks," in *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. Cham, Switzerland: Springer, 2020, pp. 508–520, doi: 10.1007/978-3-030-47358-7_52.
[32] K. Maharana, S. Mondal, and B. Nemade, "A review: Data pre-processing and data augmentation techniques," *Global Transitions Proc.*, vol. 3, no. 1, pp. 91–99, Jun. 2022, doi: 10.1016/j.gltp.2022.04.020.
[33] P. Cerda, G. Varoquaux, and B. Kégl, "Similarity encoding for learning with dirty categorical variables," *Mach. Learn.*, vol. 107, nos. 8–10, pp. 1477–1494, Sep. 2018, doi: 10.1007/s10994-018-5724-2.
[34] V. V. Starovoitov and Y. I. Golub, "Data normalization in machine learning," *Informatics*, vol. 18, no. 3, pp. 83–96, Sep. 2021, doi: 10.37661/1816-0301-2021-18-3-83-96.
[35] C. Feng, H. Wang, N. Lu, T. Chen, H. He, Y. Lu, and X. M. Tu, "Log-transformation and its implications for data analysis," *Shanghai Arch. Psychiatry*, vol. 26, no. 2, pp. 105–109, 2014, doi: 10.3969/j.issn.1002-0829.2014.02.009. [Online]. Available: https://shanghaiarchivesofpsychiatry.org/en/2014.02.009.html
[36] M. Ringnér, "What is principal component analysis?" *Nature Biotechnol.*, vol. 26, no. 3, pp. 303–304, Mar. 2008, doi: 10.1038/nbt0308-303.
[37] B.-K. Bao, G. Liu, C. Xu, and S. Yan, "Inductive robust principal component analysis," *IEEE Trans. Image Process.*, vol. 21, no. 8, pp. 3794–3800, Aug. 2012, doi: 10.1109/TIP.2012.2192742.
[38] H. Fang, P. Tang, and H. Si, "Feature selections using minimal redundancy maximal relevance algorithm for human activity recognition in smart home environments," *J. Healthcare Eng.*, vol. 2020, pp. 1–13, Nov. 2020, doi: 10.1155/2020/8876782.
[39] Z. Zhao, R. Anand, and M. Wang, "Maximum relevance and minimum redundancy feature selection methods for a marketing machine learning platform," in *Proc. IEEE Int. Conf. Data Sci. Adv. Analytics (DSAA)*, Oct. 2019, pp. 442–452, doi: 10.1109/DSAA.2019.00059.
[40] S. Uddin, A. Khan, M. E. Hossain, and M. A. Moni, "Comparing different supervised machine learning algorithms for disease prediction," *BMC Med. Informat. Decis. Making*, vol. 19, no. 1, pp. 1–16, Dec. 2019, doi: 10.1186/s12911-019-1004-8.
[41] P. Jain, M. T. Islam, and A. S. Alshammari, "Comparative analysis of machine learning techniques for metamaterial absorber performance in terahertz applications," *Alexandria Eng. J.*, vol. 103, pp. 51–59, Sep. 2024, doi: 10.1016/j.aej.2024.05.111.
[42] K. Rawal, P. D. Devendrabhai, P. Pataniya, P. Jain, A. Joshi, G. K. Solanki, and M. Tannarana, "Versatile photo-sensing ability of paper based flexible 2D-Sb0.3Sn0.7Se2 photodetector and performance prediction with machine learning algorithm," *Opt. Mater.*, vol. 152, Jun. 2024, Art. no. 115547, doi: 10.1016/j.optmat.2024.115547.
[43] A. R. Bagasta, Z. Rustam, J. Pandelaki, and W. A. Nugroho, "Comparison of cubic SVM with Gaussian SVM: Classification of infarction for detecting ischemic stroke," *IOP Conf. Ser., Mater. Sci. Eng.*, vol. 546, Sep. 2019, Art. no. 052016, doi: 10.1088/1757-899X/546/5/052016.
[44] N. Mohd Hatta, Z. Ali Shah, and S. Kasim, "Evaluate the performance of SVM kernel functions for multiclass cancer classification," *Int. J. Data Sci.*, vol. 1, no. 1, pp. 37–41, May 2020, doi: 10.18517/ijods.1.1.37-41.2020.
[45] R. A. de Paula, L. Marim, R. A. Penchel, Y. R. R. Bustamante, M. L. F. Abbade, G. Perez-Sanchez, and I. Aldaya, "Mitigation of nonlinear phase noise in single-channel coherent 16-QAM systems employing logistic regression," *Opt. Quantum Electron.*, vol. 53, no. 9, pp. 1–14, Sep. 2021, doi: 10.1007/s11082-021-03149-7.
[46] T. Pattnaik, P. Kanungo, P. K. Sahoo, T. Kar, P. Jain, M. S. Soliman, and M. T. Islam, "An efficient low complex-functional link artificial neural network-based framework for uneven light image thresholding," *IEEE Access*, vol. 12, pp. 118315–118338, 2024, doi: 10.1109/access.2024.3447716.
[47] P. Jain, J. Yedukondalu, H. Chhabra, U. Chauhan, and L. D. Sharma, "EEG-based detection of cognitive load using VMD and LightGBM classifier," *Int. J. Mach. Learn. Cybern.*, vol. 15, no. 9, pp. 4193–4210, Sep. 2024, doi: 10.1007/s13042-024-02142-2.
[48] M. Bhavsar, K. Roy, J. Kelly, and O. Olusola, "Anomaly-based intrusion detection system for IoT application," *Discover Internet Things*, vol. 3, no. 1, p. 5, May 2023, doi: 10.1007/s43926-023-00034-5.

AHMAD HOUKAN received the B.Tech. degree (Hons.) in electrical drive engineering from Aleppo University, Syria, and the master's degree (Hons.) from Siksha 'O' Anusandhan Deemed to be University, Bhubaneswar, India. He was a Teaching Assistant at the University of Aleppo for a year. Subsequently, he was sent to India to pursue the master's degree. He is currently a full-time Ph.D. Researcher in intrusion detection systems, industrial control systems (ICS), and the Industrial Internet of Things (IIoT) at C. V. Raman Global University, Bhubaneswar. He received the Al-Basil Award for Excellence three times.

ASHWIN KUMAR SAHOO is currently the Dean and a Professor with the Department of Electrical Engineering, C. V. Raman Global University, Bhubaneswar, Odisha, India. He has 27 years of teaching and research experience and two years of Industrial experience. His research interests include FACTS, microgrids, automation, AI techniques for condition monitoring, and renewable energy.

SARADA PRASAD GOCHHAYAT received the B.Tech. degree in ECE from ITER, SOA University, Bhubaneswar, the M.Tech. degree in signal processing from IIT Guwahati, and the Ph.D. degree in communication engineering from Indian Institute of Science, Bengaluru. He has past working experience as a Faculty Member at Villanova University, Villanova, PA, USA; Old Dominion University, Norfolk, VA, USA; Manipal University, India; and CGU, Bhubaneswar, India; and a Postdoctoral Research Experience at the University of Padua, Italy; and the Virginia Modeling, Analysis, and Simulation Center, Suffolk, VA, USA. He is currently an Assistant Professor at Indian Institute of Technology–Jammu, India. His research interests include blockchain, privacy in cloud computing, privacy-preserving data analytics, security and privacy in healthcare systems, and security in the IoT and 5G networks.

PRABODH KUMAR SAHOO (Member, IEEE) received the M.E. degree from Rajiv Gandhi Proudyogiki Vishwavidyalaya, Bhopal, India, in 2005, and the Ph.D. degree from the Centurion University of Technology and Management, Odisha, India, in 2019. He is currently an Associate Professor with the Department of Mechatronics, Parul University, Vadodara, Gujarat, India. He has contributed significantly to his field, having published nine peer-reviewed journal articles, presented five international conference papers, and secured two international patents and one Indian patent energy. His research interests include image processing, computer vision, and cyber-physical systems.

HAIPENG LIU received the B.Eng. and M.Eng. degrees in electrical engineering and biomedical engineering from Zhejiang University, Hangzhou, China, and the Ph.D. degree in medical sciences from The Chinese University of Hong Kong, Hong Kong, China. He was a Research Fellow with Anglia Ruskin University, U.K., from 2019 to 2020, and Coventry University, U.K., from 2020 to 2024. He is currently an Assistant Professor with Coventry University. He is the author of more than 80 journal articles, 12 conference papers, and 16 book chapters. He has edited three books on artificial intelligence in modern healthcare. He contributed to reputed publishers, including IEEE, Elsevier, and Springer, as an editorial member of five academic journals and a peer-reviewer of more than 100 papers. His research interests include biomechanics, physiological measurement, and simulation of cardiovascular diseases. He was a recipient of the Travel Award of the British Heart Foundation at the 14th International Symposium on Biomechanics in Vascular Biology and Cardiovascular Disease.

SYED GHUFRAN KHALID received the B.S. degree in biomedical engineering from the Sir Syed University of Engineering and Technology, Pakistan, the M.Sc. degree in medical engineering with specializations in medical imaging and bio instrumentation from the KTH Royal Institute of Technology, Sweden, and the Ph.D. degree in medical engineering from Anglia Ruskin University, U.K., focusing on innovative research in cuffless blood pressure estimation using photoplethysmography and machine learning. Currently, he serves as the Course Lead and a M.Sc. Lecturer in communications engineering at Nottingham Trent University. He is also a Distinguished Biomedical Engineer and an Academic with extensive expertise in electronic engineering. His research projects, which include the development of wearable medical technologies and biomedical engineering solutions, have garnered multiple travel awards and an Innovate UK grant. He has presented and published his work at international conferences and prestigious journals

PRINCE JAIN received the Doctor of Philosophy (Ph.D.) degree from Punjab Engineering College (Deemed to be University), Chandigarh, India. He is currently an Assistant Professor (Research Cadre) with the Mechatronics Engineering Department, Parul Institute of Technology, Parul University, Vadodara, India. He received the Visvesvaraya Ph.D. Scheme Fellowship to complete the Doctor of Philosophy (Ph.D.) Dissertation. He is the author and coauthor of about 60 research articles and a few book chapters on various topics related to machine learning and metamaterials. He has contributed as a peer reviewer for prestigious publishers, including IEEE, Elsevier, Springer, IOPscience, Wiley, MDPI, Frontiers, PIER, Emerald, Bentham Science, and PLOS. His research interests include machine learning, artificial intelligence, optimization techniques, metamaterial absorbers/antennas at RF, THz, and visible frequencies, material science, nanotechnology, and biomedical signal processing. He currently serves as an Academic Editor for *Scientific Reports* (Nature), *Journal of Electrical and Computer Engineering* (Wiley), *PLOS ONE*, and *Discover Applied Sciences* (Springer). He is also serving as a Topical Advisory Panel Member for *Micromachines* and *Materials* (MDPI). He is also working on an Extra-Mural Research Project sanctioned under the CSIR-ASPIRE scheme.