International Journal of Engineering Trends and Technology
ISSN: 2231-5381/ https://doi.org/10.14445/22315381/IJETT-V74I7P106

Volume 74 Issue 7, 69-95, July 2026

Original Article

An Integrated Approach to Detect Attacks in Cloud Environment using Wavelet Transforms and Cubic Kernel Classifiers

S. Divya[1], D. Kavitha[2]

[1,2]*Department of Computer Application, St Peter's Institute of Higher Education and Research, Avadi, Chennai, Tamil Nadu, India.*

[1]*Corresponding Author : divya1995.bsc@gmail.com*

Received: 29 August 2025 Revised: 08 June 2026 Accepted: 18 June 2026 Published: 28 July 2026

Abstract - *Cyber-attacks on the Internet of Things are increasingly common due to low-power devices. Effective detection algorithms are crucial for distinguishing between benign and malicious behaviour in IoT networks and video surveillance systems. These algorithms analyse network traffic patterns and detect cyber threats. However, using feature extraction-based algorithms to predict different characteristics of network traffic poses challenges in improving classifier accuracy. This paper presents a feature extraction-based cubic kernel classifier designed to detect three major attacks: Botnet, Denial of Service (DoS), and Man-in-the-Middle (MiM) that disrupt normal operations in IoT and communication networks. In our proposed method, (i) Features are extracted using three wavelet transforms: Discrete Wavelet Transform (DWT), Stationary Wavelet Transform (SWT), Transverse Quadratic Wavelet Transform (TQWT). These wavelet transforms outperform traditional methods in detecting anomalies and patterns due to their ability to analyse both time and frequency localization (ii) Highly significant features are selected using Pearson's correlation coefficient, and (iii) The selected features are classified as benign or malicious using Cubic Support Vector Machine (SVM), Cubic K-Nearest Neighbors (KNN). The use of a cubic kernel function effectively models complex patterns and non-linear relationships within the network and improves accuracy compared to traditional classifiers. Experimental results from Kitsune attack datasets show that the proposed TQWT-based cubic SVM (TCSVM) outperforms other methods, achieving 99.5% accuracy for both Botnet and DoS attacks, and 99% for the MiM attack.*

Keywords - *Cloud computing, Cyber-attacks, Wavelet transforms, Kernel classifiers, Cyber systems, Internet of Things.*

1. Introduction

Cloud computing is used in cyber systems, Internet of Things (IoT), and algorithms utilize resources without the need for direct management by the user. It provides significant flexibility and scalability to the users with applications in almost all fields, such as surveillance, smart cities, and transport systems that depend on a cloud platform for storing and retrieving information from anywhere in the world, in contrast to the local environment [1]. Cloud computing services are Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS).

Cloud computing reflects a remarkable transition from traditional client/server models to a dynamic ecosystem that supports modern business operations [1]. The cloud computing is based on time-sharing; multiple persons access one mainframe computer remotely. Then the idea of distributed computing emerged in the 1970s, enabling multiple computers to work together and share resources, which laid the groundwork for future cloud models. The term "cloud" began to be used metaphorically in networking and telecommunications in the 1990s. In 2002, Amazon Web Services (AWS) opened for business, providing storage and computing services. NASA and Rackspace created OpenStack in 2008 as a way for people to work together on cloud technologies. With the release of Microsoft Azure and big moves by companies like IBM and Oracle, cloud usage went through the roof in the 2010s. This was especially true during the global pandemic in 2020, which speed up the move to cloud solutions for remote work [2]. Cloud computing's popularity has grown due to its low cost, scalability, and ability to accommodate remote work situations. However, this change brings new security problems that organizations must solve in order to protect sensitive data and ensure regulatory compliance [2]. Misconfigurations are a leading cause of cloud security incidents, often stemming from a lack of awareness during cloud service setup, resulting in issues such as open ports and unsecured storage, as demonstrated by the McGraw-Hill breach, which exposed 22 TB of sensitive data due to a misconfigured AWS S3 bucket.

Poor access management is another key vulnerability, with weak Identity and Access Management (IAM) procedures resulting in unauthorized access; according to Verizon's research, IAM vulnerabilities accounted for 61% of analyzed breaches. Unsecured APIs pose additional risks, as evidenced by the Optus data breach, in which poor authentication methods permitted unauthorized access to sensitive customer records. Insider threats complicate security even more since individuals with legitimate access might compromise systems, either intentionally or negligently. Furthermore, the lack of encryption for sensitive data kept in the cloud can expose organizations to data breaches, while account hijacking via phishing assaults allows unauthorized access to cloud services. Cloud services are also subject to Denial-of-Service assaults, cause downtime, interrupt business operations. Server-Side Request Forgery (SSRF) vulnerabilities can arise when applications disclose internal endpoints, allowing attackers to obtain access to the cloud infrastructure [3]. Figure 1 displays the sources of the possibility of attacks in cloud computing.

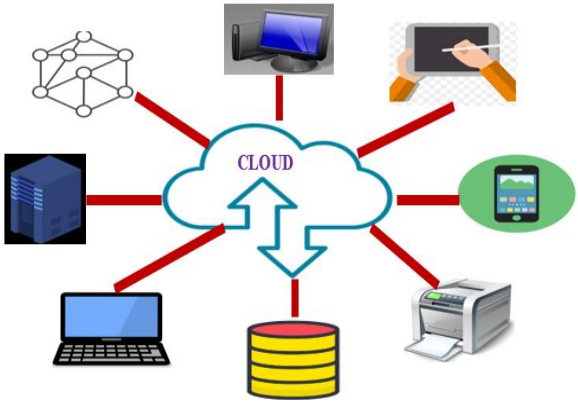


Fig. 1 Source devices of cloud attacks in real environment

Cloud applications are subject to a variety of assaults, each needing unique detection strategies. Account hijacking enables unauthorised access to cloud accounts, and detection entails using anomaly detection systems to flag suspicious login attempts and implementing Multi-Factor Authentication (MFA). DoS and Distributed Denial-of-Service (DDoS) attacks flood systems with excessive traffic, identified by anomalous spikes, applying rate limiting, and utilising DDoS protection services [3]. Cloud malware injection, which uses malicious software to undermine data integrity, can be monitored using Endpoint Detection and Response (EDR) tools and vulnerability scans on a regular basis. Insider threats occur when employees abuse their access; detecting these requires User Behaviour Analytics (UBA) to discover irregularities in activities [4]. Side-channel attacks exploit information leakage caused by system implementation, necessitating hardware security measures. This paper addresses three significant attacks, namely MiM, Botnet, and DoS, that threaten the cloud environment [3]. Figure 2 shows the occurrence of these attacks. A botnet network exploits vulnerabilities in devices to execute various malicious activities, such as sending spam, stealing data, and launching DDoS attacks [5]. Botnets can connect to a command-and-control server for instructions and may use centralized or decentralized models, with newer versions employing peer-to-peer architectures for detection [6]. MiM attacks involve intercepting communications between two parties that are directly connected, often through techniques like eavesdropping on unsecured Wi-Fi networks or phishing to trick users into revealing their credentials. The consequences of MiM attacks can be severe, leading to data theft and financial losses [3]. DoS attacks aim to disrupt network services by overwhelming them with traffic, causing operational disruptions and significant financial impacts. These attacks can flood targets with excessive requests or exploit network protocol vulnerabilities, and when executed via multiple compromised systems, they become DDoS attacks [5]. Data availability is another area where there is a possibility of more attacks threatening, as it includes the accessibility and dependability of data stored in cloud environments [3]. Organisations rely on constant access to their data for operational efficiency, decision-making, and customer service, but various issues can pose challenges to data availability. Failures on the host server or storage might make data inaccessible, especially in systems without redundancy safeguards. Network failures also threaten access, as outages can prevent users from reaching cloud-stored data, leading to significant service disruptions. Furthermore, poor data quality, characterized by inconsistencies, incompleteness, or redundancy, affects effective decision-making even when the data is available. Compatibility concerns in hybrid setups further complicate the access, where data functions well in one system may not be usable in another. Cyberattacks, such as ransomware, can encrypt or erase data, rendering it unavailable to legitimate users. Slow data transfers, which are frequently caused by latency or bandwidth constraints, can also restrict access, particularly for large datasets. Regularly monitor data quality, implement disaster recovery plans, and take advantage of cloud provider features that include built-in redundancy and high availability,

which enhance data availability in the cloud. This paper proposes a strategy to detect and classify Botnet, DoS, and MiM attacks as benign or malicious.

1.1. Research Gap

Though several algorithms exist in the literature to detect and classify attacks, it is critical to analyse the traffic in the network and detect before the occurrence of damage [6]. Few algorithms proposed in the literature detect attacks early, but the continuous derivation of new attack patterns makes the detection difficult and increases the error rate. Further datasets are not publicly available, and most of the recent models are validated with simulated data, which restricts the detailed analysis of newly developed models [15]. Table 1 shows the research gap and novelty.

Table 1. Research gap with problem statement and novelty

Research Gap	Problem	Proposed Novelty
Existing models poorly adapt to emerging cyberattacks	Dynamic and nonlinear attack patterns	Cubic kernel-based CSVM and CKNN classifiers
High false detection due to irrelevant traffic features	High-dimensional network traffic	Pearson correlation-based feature optimization
Limited preprocessing for abrupt traffic changes	Abrupt traffic variations during attacks	DWT, SWT, and TQWT preprocessing techniques
Lower accuracy in real-time attack detection	Low detection accuracy in existing models	Hybrid wavelet-ML integrated detection framework
Existing methods focus on single attack categories	Multi-attack detection complexity	Unified detection of Botnet, DoS, and MiM attacks

1.2. Problem Statement

The need for cyberattack detection algorithms to differentiate between benign and malicious behaviour is difficult in the digital landscape, particularly within IoT networks and video surveillance systems. Detection algorithms play a vital role in identifying and mitigating cyber threats by analyzing network traffic patterns to detect anomalies that may indicate malicious activities [19]. Machine learning techniques enhance detection accuracy and are designed for specific types of attacks only. By employing proactive frameworks that utilize machine learning and classification models, early detection of potential threats is possible, thereby preventing data breaches and system disruptions. Also, these algorithms easily adapt to evolving attack methodologies, ensuring ongoing protection against new and sophisticated cyber threats [15]. The implementation of effective detection systems not only safeguards sensitive information but also optimizes resource management within constrained environments like IoT, where devices often have limited processing power. In summary, robust cyberattack detection algorithms are essential for the security and integrity of networked systems nowadays, where cyber threats are becoming more prevalent and sophisticated. A challenge, such as automatic detection, prevention, and mitigation of attacks with high accuracy and enhanced methods in each stage of the whole process, is needed in a cloud environment [27]. This paper proposes a detection algorithm by integrating a wavelet transform-based pre-processing technique and machine learning classifier models to differentiate benign and malicious behaviour in Botnet, DoS, and MiM-affected network dataset captured from a video surveillance system and from an IoT network.

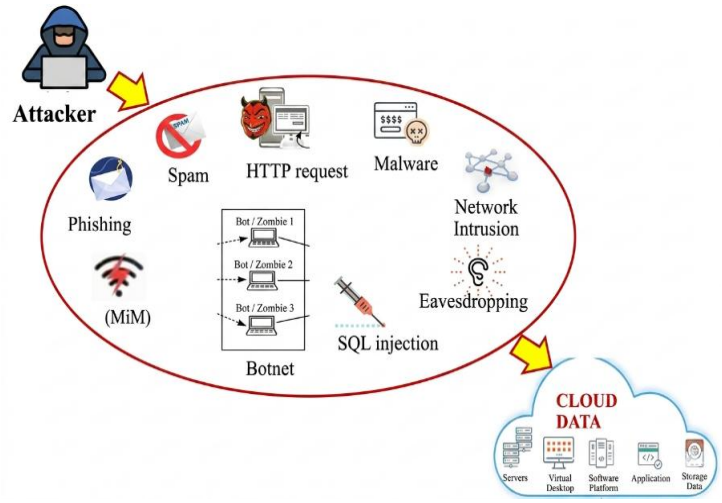


Fig. 2 Various attacks in a cloud environment

1.3. Contributions

Attacks create more volume and changes in the data stored in the cloud. Therefore, the large data needs to be pre-processed, and highly significant features need to be selected to analyse the effects caused by attacks. The proposed methodology integrates wavelet transforms and a cubic kernel classifier to differentiate benign and malicious behaviour in a cloud environment. The major contributions of the proposed work are:

- To pre-process the data to detect abrupt behavioural changes in the performance of the network during data transmission using DWT, SWT, and TQWT.
- To select highly significant features that shows abrupt variations across the traffic network from statistical measures of pre-processed data using the Pearson co-efficient of correlation.
- To improve detection accuracy by creating complex decision boundaries that distinguish between benign and malicious behaviour in non-linear datasets using cubic kernel-based classifiers, namely Cubic Support Vector Machine (CSVM) and Cubic K-Nearest Neighbors (CKNN).
- To propose a feature extraction-based classifier for the detection of benign and malicious behaviour in three types of attacks, namely (i) Botnet, (ii) DoS, and (iii) MiM.

2. Related Work

The proposed work combines feature extraction and a kernel classifier to detect benign and malicious behaviour in MiM, Botnet, and DoS attacks. This section details the recent studies related to the proposed work. Ashley Woodiss-Field et al., conducted experiments using several established botnet detection techniques, namely BotMiner, BotProbe, and BotHunter. These techniques are evaluated on IoT networks. Multiple scenarios were tested. Additionally, externally sourced datasets are evaluated and validated for each detection method [20]. Shamshair Ali et al., introduce an innovative hybrid deep learning model that combines LSTM Autoencoders and Multilayer Perceptrons to detect botnets. Integrating these technologies allows the model to effectively identify complex botnet behaviors in IoT environments [21]. Swapna Thota et al., developed a deep learning-based botnet detection algorithm known as DenseNet - Binary Moth Flame Optimization (DenseNet-BMFO). DenseNet- BMFO is superior in performance compared to other algorithms. Simulation results indicate that DenseNet-BMFO is both reliable and effective in identifying IoT network intrusion threats, achieving an impressive accuracy rate of 98.25% [23]. SK Khaja Shareef et al., introduces Zebra Optimization Algorithm (ZOA) for feature selection, and classification is done with a Dual-channel GAN. DGAN employs Node Attention Networks and Semantic Attention Networks to capture complex interactions and detect Botnet. [22]. Mohammad Al-Fawa'reh et al., proposed a MalBoT-DRL method for botnet detection. MalBoT-DRL improved generalizability and robust resilience against drift. [24]. Alvaro Michelena proposed an innovative hybrid approach for developing Intrusion Detection System (IDS), based model for detecting attacks. The first stage focuses on feature extraction, while the second stage employs various supervised classification techniques, all applied to a dataset derived from Message Queuing Telemetry Transport (MQTT). The most effective results are achieved using the method with the highest computational cost. This advancement enables a functional IDS to effectively prevent MQTT attacks [11]. Morgan Reece et al., present a multi-cloud victim web application structured as component services, deployed across various cloud service providers. This distribution across multiple providers increases the attack surface of the multi-cloud victim web application. Using ParrotOS as our exploitation server, an attack is conducted on an application hosted on three distinct cloud platforms: AWS, Azure, and Rackspace, and its effectiveness is validated [12].

Rajakumaran Gayathri et al., integrated Network Management Protocol, Kullback-Leibler Distance, Access Control Rules, and Moving Target Defense. Performance is validated by implementing the model on the Amazon Web Services (AWS) platform [3]. Sivasankari and Kamalakannan introduced a Regression technique to detect and mitigate attacks and to ensure an attack-free path from the source to the destination within an IoT network. The performance of algorithms is analyzed across various metrics, and concluded that Gaussian Process Regression achieves a higher detection rate for attacks while minimizing the misclassification rate [32]. Hamidreza Fereidouni et al., highlight issues related to IoT security and investigate future strategies and technologies aimed at enhancing detection and prevention mechanisms against MitM attacks [4]. Asha Varma Song and Ganesh Reddy Karr presented a more efficient integrated Software-Defined Networking (SDN) framework to detect DDoS traffic patterns within an SDN-based cloud. It employs Recursive Feature Elimination, Density-Based Spatial Clustering, and time series methods, detection time is 20 seconds [13]. Gottapu Sankara Rao and Krishna Subbarao proposed a Deep Learning model that develops both binary and multiclass classification systems capable of differentiating between network attack activities and normal traffic. During detection, the system assesses whether the data is indicative of an attack or regular network activity. The Multi-Layer Perceptron (MLP) algorithm enables the model to identify threats with a success rate of 97%. This paper analyzes traffic behaviour and employs traffic filtering techniques to remove suspicious or attack-signature traffic and focuses on network DoS/DDoS attacks and their prevention, employing strategies to detect and mitigate flood-based DoS attacks to ensure system functionality and network security [17]. Mahdi Nsaif Jasim and Methaq Talib Gaata proposed a method to detect DDoS using the K-Means clustering algorithm. The proposed semi-supervised approach is trained and tested with the

CICIDS2017 dataset. Hybrid feature selection methods and multiple training, testing datasets are used to detect DDoS using optimum 2 centroids generated, and is used to classify network traffic [14]. Lal Mohan Pattnaik et al., proposed a model that utilizes data fusion applications for secure cloud services and to detect DDoS attacks through machine learning classifiers. Decision Trees, Naive Bayes, SVM, and KNN are implemented and classify DDoS attacks. The experimental results indicate that the Decision Tree model is the most effective and reliable method for classifying cloud DDoS attacks [16]. Haya Alubaidan et al., detected DDoS attacks in Software-Defined Networks. Machine learning algorithms are combined with a feature selection technique to identify DDoS attacks within network traffic. The results of this experiment highlight justify the use of feature selection to enhance performance. Finally, the Random Forest classifier used to result in an accuracy of 0.99 in detecting DDoS attacks [15]. Various methods in the literature and their results are analyzed in brief in Table 2. Table 3 shows a literature review comparison with recent articles.

Table 2. Existing methods in the literature

Ref. No	Reference	Technique Name	Discussion
1	Ashley Woodiss-Field et al.,[20]	BotMiner, BotProbe, and BotHunter	The number of false positives needs to be reduced in both BotProbe and BotMiner approaches & average accuracy of 98% achieved in all the methods.
2	Shamshair et al.,[21]	LSTM Autoencoders and Multilayer Perceptrons	Optimization techniques can be used to increase the effectiveness & accuracy rate of 99.77% and 99.67% is achieved.
3	Swapna Thota et al.,[23]	Binary Moth Flame Optimization (DenseNet-BMFO).	Early detection stage needs to include for further effectiveness, & 98.25% accuracy is achieved.
4	SK Khaja Shareef et al., [22]	Graph Attention Network (GAN	Method to be tested for real-time detection and adaptation to various changes in botnet behaviour, and achieved an accuracy rate of 99.87%.
5	Mohammad Al-Fawa'reh et al.,[24]	malware botnet detection with reinforcement learning	Tested on two datasets and needs to be extended for more datasets & high accuracy rate late detection phase. Adversarial attacks need to be examined.
6	Alvaro Michelena [11]	Hybrid Intrusion Detection System	Proposed method to be tested for different dimension reduction and feature extraction techniques.
7	Rajakumaran Gayathri et al., [3]	Simple Network Management Protocol, Kullback-Leibler Distance	Need to be extended for other types of attacks.
8	Sivasankari and Kamalakannan [32]	Regression algorithms	Gaussian Process Regression achieves a higher detection rate for attacks and needs to be extended as a multiple regression problem.
9	Asha Varma Song and Ganesh Reddy Karr [13]	Recursive Feature Elimination, Density-Based Spatial Clustering	Training and testing data need to be tuned further, & 99.92% detection rate in 20 seconds.
10	Gottapu Sankara Rao and Krishna Subbarao [17]	Multi-layer Perceptron	Differentiates network attack traffic and normal traffic better than existing and has been tested on big networks with high security protocols, and to mitigate the attacks & 97% accuracy.
11	Mahdi Nsaif Jasim and Methaq Talib Gaata [14]	K-Means clustering algorithm	Tested for a single dataset, needs to be extended for more datasets, and more machine learning algorithms need to be utilized, & 99 %accuracy is obtained
12	Lal Mohan Pattnaik et al.,[16]	Decision Trees.	A decision tree classifier is best compared to other existing classifiers. Applicable only for single attack DoS, and to be extended for modern attacks, & 99 %accuracy is obtained.
13	Haya Alubaidan et al., [15]	SDN and Machine learning	Feature extraction can be modified for better performance, & 99% accuracy is achieved.
14	Proposed method	Feature extraction based cubic kernel classifiers	Transverse quadratic wavelet Transform Based Cubic SVM (TCSVM) performs better to detect Botnet, DoS, and MiM attacks.

Table 3. Literature review comparison with recent articles

Threat Category	Attack Type	Method Used	Pipeline Type	Targeted Network Component	Reference / DOI
Botnet Attack	Mirai Botnet	Hybrid Deep Learning	Attack-Specific Pipeline	IoT Devices and Cloud Servers	Ali et al., Alexandria Engineering Journal, 2024, DOI: 10.1016/j.aej.2024.05.113 [21]
DoS Attack	Flooding / DDoS Attack	Systematic IDS and Traffic Analysis	Attack-Specific Pipeline	Network Servers and Gateways	Gelgi et al., Sensors, 2024, DOI: 10.3390/s24113571 [33]
MiM Attack	Zero-Day Botnet Attack	Federated Deep Learning	Distributed Detection Pipeline	Edge and Federated IoT Devices	Jogunola et al., IEEE IoT Journal, DOI: 10.1109/JIOT.2021.3100755 [34]
Proposed Unified Framework	Botnet + DoS + MiM	TQWT-Based Cubic SVM (TCSVM)	Single Unified Detection Pipeline	IoT, Cloud, and Surveillance Networks	Proposed Work

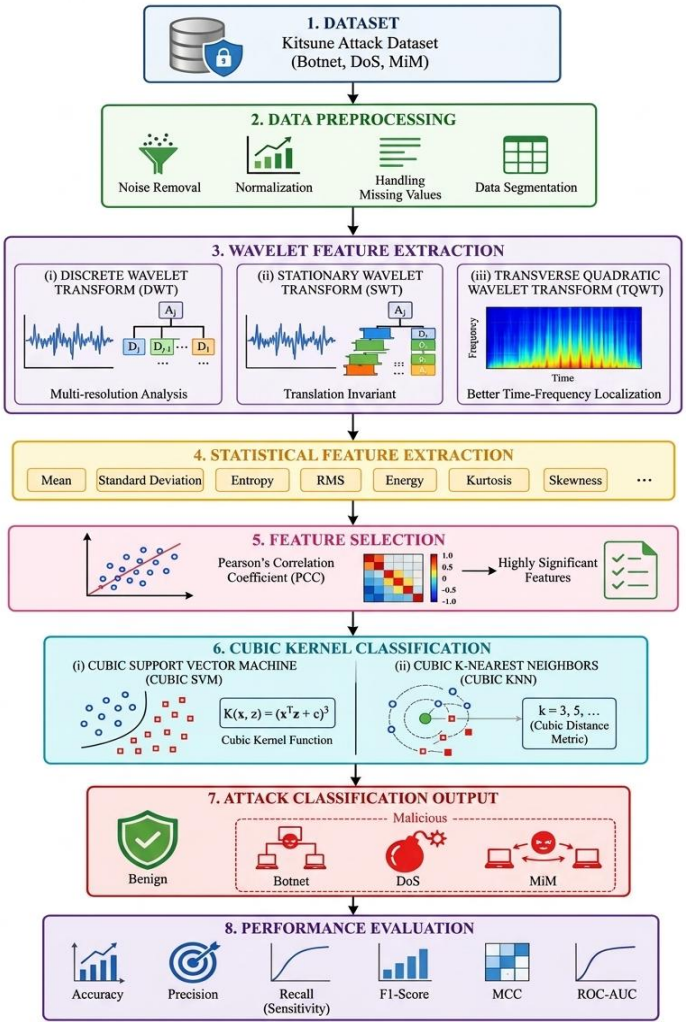


Fig. 3 Proposed framework

In summary, the integrated method of feature extraction and classifier improves the overall performance of attack detection compared with existing methods.

3. Methodology

Proposed methodology is shown in Figure 3.This section describes the proposed work, techniques, and dataset utilized

in detail. Datasets are pre-processed using wavelet transforms, and highly significant features are selected using Pearson's co-efficient of correlation. Features selected are fed as input to Cubic kernel classifiers and are differentiated as benign and malicious behaviour. A few of cloud cyber-attacks that target vulnerabilities in cloud infrastructure and applications are account hijacking, malware injection, misconfigurations, and insider threats. The objective of the proposed work is to implement a system to detect MiM, Botnet, and DoS attacks and classify as benign or malicious in the cloud environment [3]. The major causes of these attacks that results in behavioural changes in the network are outlined in Figure 4.

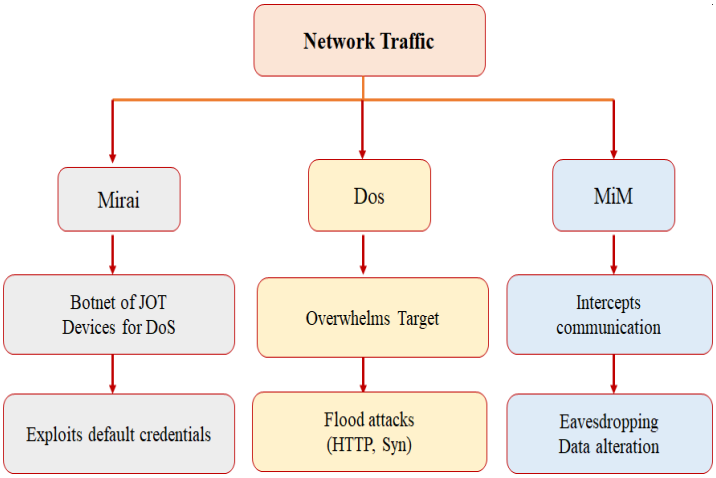


Fig. 4 Causes of Botnet, DoS, and MiM attacks in network traffic

3.1. Input Data

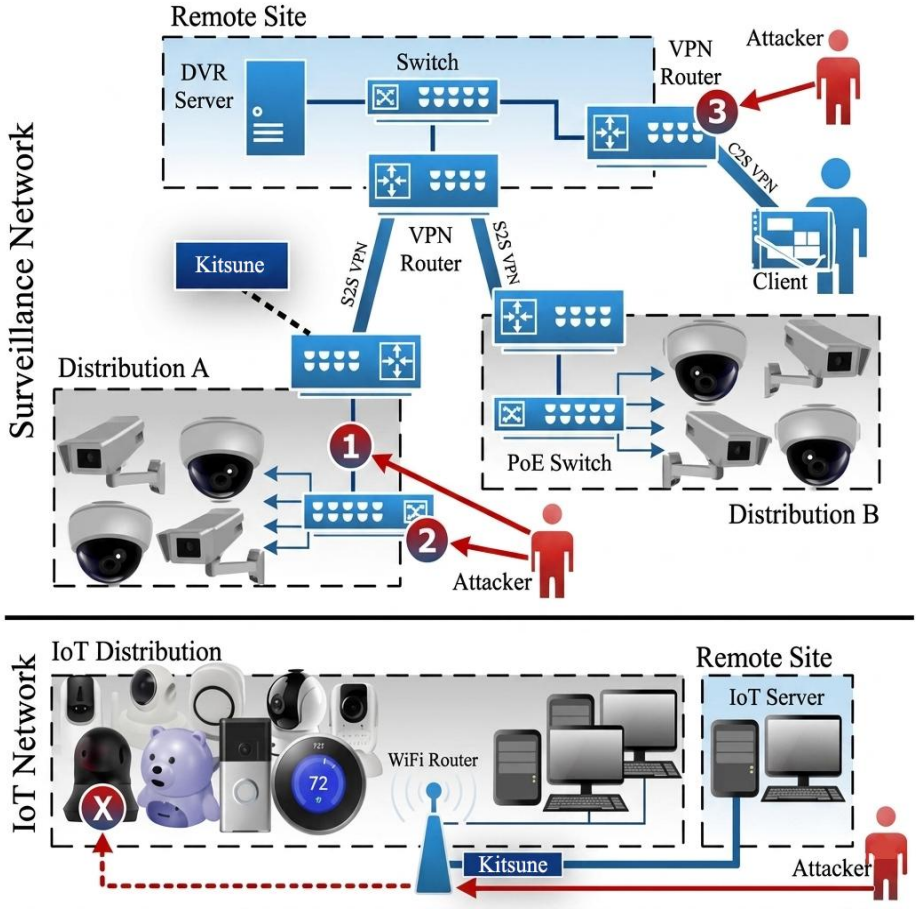


Fig. 5 Attack dataset generation in video surveillance and IoT network [19]

The Kitune network attack dataset is employed to evaluate the proposed methodology [Figure 5] [19]. Each dataset comprises millions of network packets and encompasses various cyberattacks.

The analysis utilizes three specific datasets: Botnet, DoS, and MiM attacks. (i) Dataset 1: The Mirai (Botnet) dataset contains approximately 700,000 samples and features 115 variables. (ii) Dataset 2: The DoS dataset includes around 800,000 samples, and (iii) Dataset 3: The MiM attack dataset consists of 700,000 samples.

To further analyse and validate the proposed methodology, several samples of varying sizes are generated from the original input data. Three cyberattacks, their effects, and their source of origin are explained

3.1.1. Botnet

A botnet generates significant volumes of traffic, often targeting legitimate user systems [23]. Detecting botnets poses considerable challenges, necessitating continuous monitoring across multiple devices. These attacks frequently occur within financial institutions, including banks, payment processors, SaaS platforms, and telecommunications companies. Botnets are critical components of modern distributed Denial of Service (DDoS) attacks due to their capacity to produce vast amounts of traffic from numerous distributed sources [24]. They exploit vulnerabilities in various devices and employ a range of attack methodologies with notable flexibility. The dynamic nature of botnets complicates detection efforts, as they can adapt to circumvent traditional security measures [21]. Figure 6 shows the causes of Botnet. Table 4 shows the dataset description.

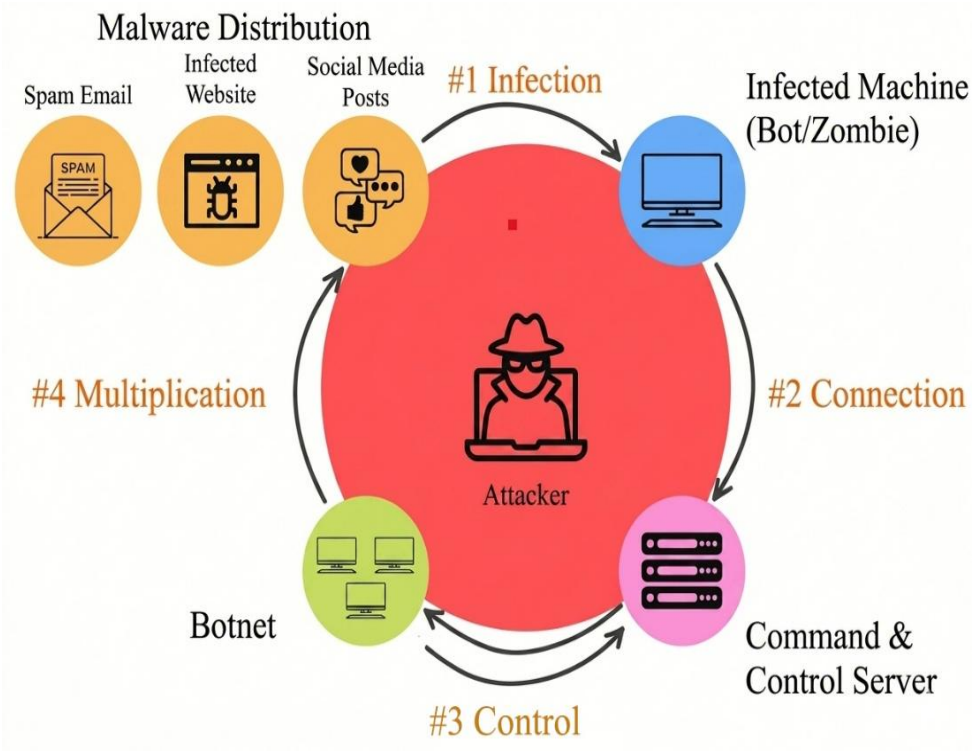


Fig. 6 Causes of botnet

Table 4. Description of dataset

Dataset / Attack Type	No. of Samples	No. of Features	Benign Traffic	Attack Traffic	Attack Type	Traffic Source	Sampling Rate	Data Format	Environment
Mirai (Botnet)	220,000	115	110,000	110,000	Botnet	IoT Devices	1 s interval	CSV / PCAP	Smart Home IoT Network
DoS	180,000	115	90,000	90,000	Denial of Service	Surveillance Network	1 s interval	CSV / PCAP	Cloud-IoT Environment
MiM	150,000	115	75,000	75,000	Man-in-the-Middle	Communication Network	1 s interval	CSV / PCAP	Video Surveillance System

3.1.2. Denial of Service (DoS)

Denial of Service (DoS) attacks are among the most prevalent forms of cyberattacks, originating from a single source to overwhelm a target with excessive traffic, rendering it unavailable [25]. Figure 7 shows the causes of Denial of Service (DoS).

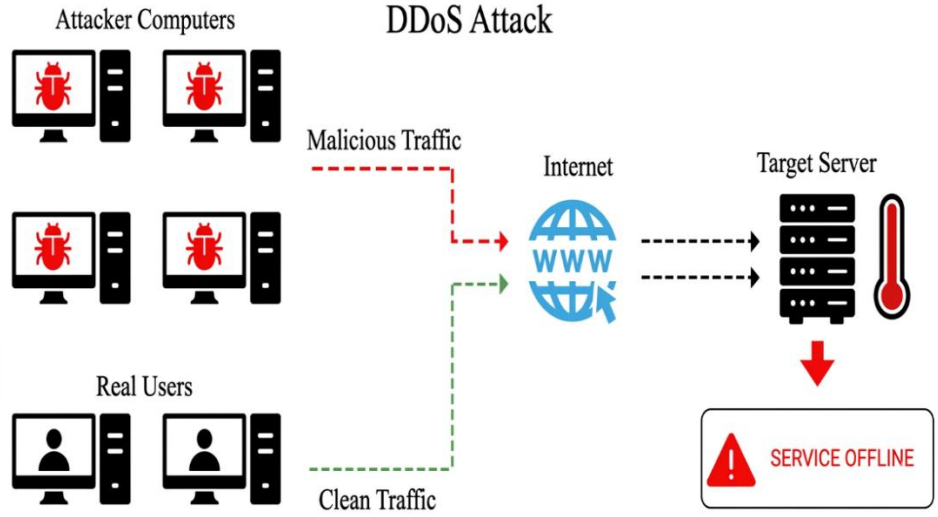


Fig. 7 Causes of Denial of Service (DoS)

These attacks can be launched from various locations, including servers, websites, services, gaming platforms, and e-commerce sites. DoS attacks are challenging to defend against due to the distributed nature of traffic. Various methods of launching DoS attacks include flood attack (sending a large volume of packets on the target system), HTTP requests that overwhelm servers, amplifier attacks which sends a request with a spoofed IP address and amplify the amount of traffic from multiple sources, and sending malformed attacks causing the target system to crash or reboot [26]. The diverse range of techniques employed in DoS attacks underscores the complexity of mitigating such threats effectively [27].

3.1.3. Man in the Middle (MiM)

MiM attacks manipulate information [4]. These attacks are in scenarios such as financial transactions, sensitive data exchanges, public Wi-Fi networks, corporate networks, healthcare systems, and other unsecured environments.

The vulnerability of these settings makes them prime targets for MiM attacks, where the attacker can eavesdrop on communications or alter the exchanged data without the knowledge of either party involved [11]. Figure 8 shows the cause of Man in the Middle (MiM)

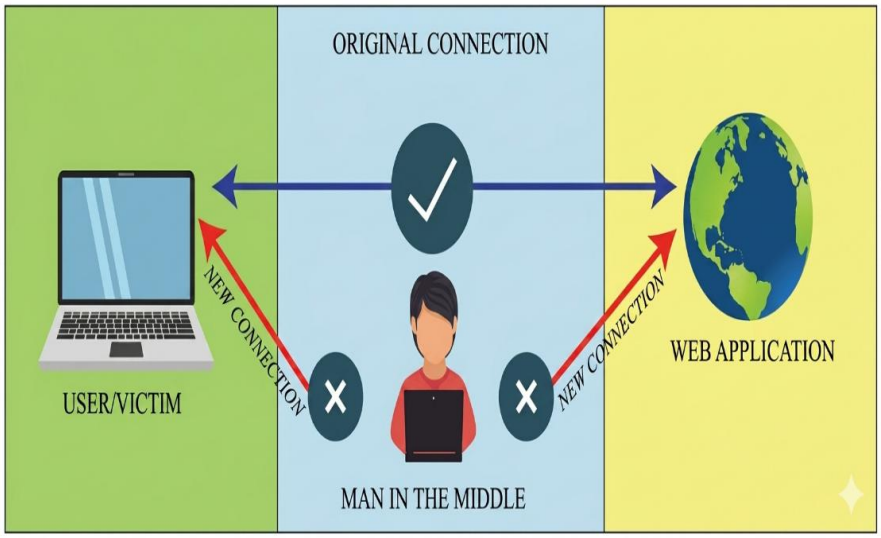


Fig. 8 Cause of Man in the Middle (MiM)

3.2. Pre-Processing

Pre-processing the data plays a primary role and enhances the post-processing classification model. The various cleaning, transformation, and feature extraction techniques improve the quality of input data, leading to more accurate predictions [9]. Feature extraction techniques improve the accuracy of post-processing in differentiating the benign and malicious activity with a reduced false positive rate. In the proposed work, DWT, SWT, and TQWT are utilized to extract features from input data. Further, the most significant features are selected and processed using Pearson's co-efficient of correlation for post-processing [8].

3.2.1. Discrete Wavelet Transform

DWT is used as a pre-processing technique in the proposed work to extract co-efficient as it effectively reduces noise and enhances the data by removing high-frequency noise, which further results in desired wavelet coefficients that improves the post-processing step faster and require only minimum memory to process input data [9]. Basically, DWT decomposes the data into different frequency components characterized by properties such as zero mean and finite energy in both time and frequency. Mathematically, DWT is the processing of convoluting input data $x[n]$ with a filter co-efficient $l[n]$ or $h[n]$ to result in a low frequency and high frequency components expressed as:

Approximation or low frequency components:

$$A[n] = \sum_k x[n] * l[n-k] \tag{1}$$

Details or high-frequency components:

$$D[n] = \sum_k x[n] * h[n-k] \tag{2}$$

By applying the low-pass filter recursively on approximation coefficients results in a multi-resolution structure for the original input data at various levels, which improves the classification accuracy of time series data. Thus, the DWT is an efficient technique used to extract features on input dataset for further processing. DWT results in a perfect reconstruction of the original input by convolving the inversion filters and wavelet co-efficient.

3.2.2. Stationary Wavelet Transform (SWT)

In contrast to DWT, SWT is more efficient in terms of translation invariance, Gibbs phenomenon, and multi-resolution analysis as it maintains the same length throughout the process of transformation. The same length is maintained in SWT by adding zeros to the wavelet co-efficients. In the inverse process, the original input is reconstructed by applying a thresholding operation on the wavelet co-efficients. SWT results in more accurate and reliable features in a non-stationary environment and improves the classification accuracy of the proposed method [9].

Mathematically, SWT for input data $x[n]$ results in approximation and detail coefficients as expressed as:

For Level 1:

Approximation Coefficients:

$$\mathrm{cA_j[n]} = \mathrm{s[n]} * \mathrm{L_o D[n]} \tag{3}$$

Detail Coefficients:

$$\mathrm{cD_j[n]} = \mathrm{s[n]} * \mathrm{H_i D[n]} \tag{4}$$

For level j:

Approximation Coefficients:

$$\mathrm{cA_{j+1}[n]} = \mathrm{cA_j[n]} * \mathrm{L_o D[n]cA_j + 1[n]} = \mathrm{cA_j[n]} * \mathrm{L_o D[n]} \tag{5}$$

Detail Coefficients:

$$\mathrm{cD_{j+1}[n]} = \mathrm{cA_j[n]} * \mathrm{H_i D[n]cD_j + 1[n]} = \mathrm{cA_j[n]} * \mathrm{H_i D[n]} \tag{6}$$

SWT on input data results in a matrix-structured output in which first row represents the detail coefficients and the last row represents the approximation co-efficient. The ability to preserve all data improves the precision of classification models that require these features. Statistical parameters were calculated from the wavelet coefficients to extract the most significant features for classification.

3.2.3. Tunable Q-factor Wavelet Transform (TQWT)

Both DWT and SWT use a single wave-like oscillation to obtain co-efficient for input data, but TQWT analyses input data with varying oscillatory behaviors [9]. Unlike traditional wavelet transforms, TQWT allows tuning its Q-factor, which measures oscillatory characteristics. This flexibility allows TQWT to predict varying characteristics of network data, making it particularly effective for feature extraction. TQWT, as a filter bank structure, utilizes two-channel filter banks to reconstruct the signal perfectly as input. The frequency response functions $\mathrm{H_0(\omega)}$ and $\mathrm{H_1(\omega)}$ employed are:

$$\mathrm{H_0(\omega)} = \left\{ \begin{matrix} 1 & \mathrm{if} |\omega| \leq (1-\alpha)\pi \\ 0 & \mathrm{otherwise} \end{matrix} \right\} \tag{7}$$

$$\mathrm{H_1(\omega)} = \left\{ \begin{matrix} 1 & \mathrm{if} |\omega| \geq (1+\alpha)\pi \\ 0 & \mathrm{otherwise} \end{matrix} \right\} \tag{8}$$

Where $'\alpha'$ is a parameter related to the bandwidth of the filter, these conditions ensure that the filters satisfy the perfect reconstruction criteria. The center frequency $\mathrm{f_c^{(j)}}$ and bandwidth $\mathrm{B_w(j)}$ For each scale, j can be calculated as:

$$f_c^{(j)} = \frac{(2-\beta)\alpha^{j-1} f_s}{4} \quad j = 1,2 \ldots J \tag{9}$$

where f_s is the sampling frequency, and α is a scaling factor that influences the spacing between sub-bands.

TQWT co-efficients partition the energy of the data into sub-bands that aids in feature extraction and enhances overall quality of the data [9]. Specifically, TQWT highlights significant trends and patterns that simplify further analysis and leads to results that are more accurate. In comparison to DWT and SWT, TQWT enhances data analysis through its customizable filtering options with the use of Q factor, perfect signal reconstruction capabilities, multi-resolution analysis support, and efficiently process large datasets. Mean, Standard deviation, and Energy across sub-bands are calculated to select the most significant features.

3.3. Feature Extraction Using Pearson's Co-efficient of Correlation

The proposed methodology employs DWT, SWT, and TQWT on input data as a pre-processing technique to extract features. Statistical parameters are calculated from wavelet co-efficients. Highly correlated variables based on these features are selected using Pearson's correlation of co-efficient [7]. Features selected are taken as input to classifier to differentiate benign and malicious activity in network data.

Pearson correlation coefficient is based on the covariance of the variables divided by the product of standard deviations, making it a normalized measure of how much two variables change together [8].

Mathematically, for two features 'x' and 'y' with their means μ_x and μ_y, the Pearson's coefficient of correlation 'r' is given by,

$$r = \frac{\sum(x-\mu_x)(y-\mu_y)}{\sqrt{(x-\mu_x)^2(y-\mu_y)^2}} \tag{10}$$

Based on the coefficient of correlation values obtained, higher-order significant variables are selected for further processing.

3.4. Classification

The pre-processed data is fed as an input to classifiers to differentiate benign and malicious activities. The primary goal of classification is to build a model that can accurately assign labels to observations based on their characteristics [8]. Two types of classification are binary and multi-class classification, output belongs to one of multiple classes such as types of flowers) [16].

Existing classification methods in the literature include logistic regression [10], decision trees [16], random forests [31], SVM [18], and KNN. Logistic regression works well for binary outcomes, while decision trees provide a straightforward visual representation of decision processes. Random forests improve predictive performance by aggregating multiple decision trees. SVMs are effective in high-dimensional spaces for separating classes with hyperplanes [18], and KNN classifies based on nearest neighbours [28]. From the literature, non-linear decision boundary-based kernel classifiers, KNN, and SVM are utilized in the proposed work to classify different types of classes in the cloud attack environment.

3.4.1. Cubic K-Nearest Neighbour (CKNN)

K-Nearest Neighbour (KNN) is an example of a supervised learning algorithm that is used for the classification of benign and malicious behaviour in the proposed work. KNN is simple, easy to implement, and helps to develop more complex, efficient algorithms. KNN identifies the benign and malicious samples by finding 'K' nearest neighbours from the training set and assigning a label based on majority vote for classification. To improve the accuracy of traditional KNN, kernel functions with weights are modelled for classification applications. Traditional KNN uses all neighbour samples to make a final decision, whereas in Kernel KNN, weights are assigned to samples that influence high compared to far away samples. Different type of Kernel in the literature improves accuracy and robustness in handling non-linear samples efficiently, includes gaussian, parabolic, triangular, cubic, and uniform kernel functions [15]. Cubic KNN is used in the proposed method to classify benign and malicious behaviours in network data. As a first step. Input data is mapped into a high-dimensional feature space to handle the complex relationship between data samples. A polynomial kernel creates interactions between features that are not linearly separable and effectively differentiates the classes. Kernel function $K(Z_i, Z_j)$ finds the inner product between the samples in the feature space defined by $K(Z_i Z_j) = \langle \varphi(Z_i), |\varphi(Z_j)\rangle$ where φ is the mapped function. The mapping function allows to find the similarity between pair of samples directly without the need of computing their coordinates in high-dimensional space [16]. The cubic kernel function used is given by,

$$K(Z_i, Z_j) = ({Z_i}^T Z_j + 1)^3 \tag{11}$$

With the use of kernels, the computation time of KNN is minimized in handling large datasets, as it transforms each sample into a new feature space. The value of 'K' in cubic KNN is fixed using cross-validation to identify the 'K' nearest neighbour for a sample, and a label is assigned based on majority vote among the neighbors.

3.4.2. Cubic Support Vector Machine (CSVM)

SVM is a supervised machine learning algorithm. In comparison to other classification algorithms, such as logistic

regression, decision trees, SVM performs better in separating linear and non-linear datasets with kernel functions. This leads to the robustness of the SVM algorithm against over-fitting in handling high-dimensional complex datasets. Also, SVM needs only a minimum memory to process the algorithm, as it uses only support vectors for classification. Further, to improve the classification accuracy, pre-processed data is fed as an input to a cubic SVM for the detection of attacks in the network. In the proposed work, Cubic SVM is employed on the wavelet transform based pre-processed data to increase the classification accuracy in comparison to other existing methods [28]. The cubic kernel function used to map the input data is given by,

$$K(Z_i, Z_j) = (Z_i{}^T Z_j + 1)^3 \quad (12)$$

The input samples denoted as 'Z' are transformed into a high-dimensional space with the use of Equation (12). As a first step, feature mapping is performed to expand the feature set, where a single feature 'Z' is mapped as,

$$\varphi(Z) = \begin{bmatrix} 1 \\ Z \\ Z^2 \\ Z^3 \end{bmatrix} \quad (13)$$

With the use of the transformation function $'\varphi'$, each sample in 'Z' is mapped as a polynomial term of third degree. The high-dimensional representation of input data allows creating complex decision boundaries that separate the non-linear samples effectively.

After mapping of input data, SVM searches for a hyperplane in high-dimensional space that maximizes the margin between different classes. The decision function that relates new samples and original samples during training can be expressed as:

$$f(Z) = \sum_{i=1}^{n} \alpha_i y_i K(Z_i, Z_j) + b \quad (14)$$

Where y_i- class labels, α_i- weights determined during the training phase and b- bias term.

The polynomial transformation of input data in cubic SVM allows cubic kernels to effectively classify as benign and malicious behaviour in traffic network data [18].

Effective modelling and computation of non-linearities with kernels offers flexibility and robustness in the classification of complex datasets. In summary, kernel-based classifiers Cubic KNN and Cubic SVM are used in the proposed work to differentiate benign and malicious behaviors in the network [31]. In both algorithms, non-linearity between samples is analyzed by a weight factor and an expanded feature set to increase the classification accuracy compared to traditional distance metrics-based classification algorithms. Though both algorithms enhance the performance of classification, they operate fundamentally in a different way on datasets.

(i) In SVM kernel-based algorithms, the kernel function converts original data into transformed data based on similarity-on-similarity value computed using the kernel function by creating non-linear boundaries between different classes. In KNN kernel algorithms, kernel function modifies the distance calculation in traditional KNN. Different weights are assigned to neighbors that influences to the original data samples. High weights assigned to the closest neighbors that plays a major role in deciding as benign or malicious. The weighting factor allows kernel KNN to handle complex datasets effectively.

(ii) Kernel function in SVM directly shows the effect on decision boundaries and make decision by maximizing classes, minimizes classification errors. Whereas the kernel function in KNN affects the weights of neighbors that plays a major role in making a decision.

(iii) Kernel SVM performs only the inner dot product instead of the matrix transform, which minimizes the computation complexity of the algorithm. Whereas traditional KNN increases the computation complexity, and the same is reduced by implementing the algorithm on reduced dimensional set or on a pre-processed dataset.

The next section discusses the performance of cubic KNN and cubic SVM algorithms in detecting DoS, MiM, Botnet attacks, and classifies as benign or malicious in network traffic data.

3.5. Discussion

The performance of the proposed work was evaluated through three different Kitsune datasets, as detailed in Table 5. The proposed methodology has been implemented in MATLAB R 2022a on intel core i5 X64 processor.

Table 5. Datasets

Dataset name	Number of samples	Original Samples
Mirai (Botnet)	10000,20000,30000	7,64,137
DoS	10000,20000,30000	2,771,276
Mim	10000,20000,30000	2,504,276

Datasets are pre-processed using the wavelet transform and features selected using Pearson's co-efficient of correlation. Pre-processed data is fed as an input to classifiers, namely cubic SVM and cubic KNN, to differentiate benign and malicious samples in a cloud environment. Six proposed methods are examined in this work, namely DWT-based cubic SVM (DCSVM), SWT-based cubic SVM (SCSVM), TQWT-based cubic SVM (TCSVM), DWT-based cubic KNN

(DCKNN), SWT-based cubic KNN (SCKNN), and TQWT-based cubic KNN (TCKNN). Performance measures evaluated and described in Table 6.

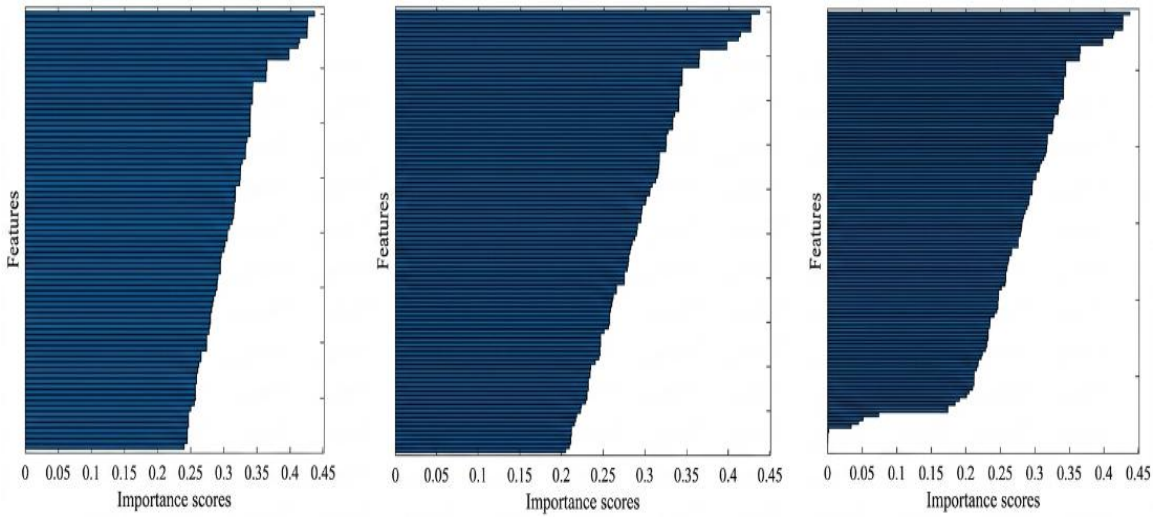
Table 6. Performance metrics

S.No	Parameters	Formula
1	Accuracy	$\frac{(TP+TN)}{(TP+TN+FP+FN)}$
2	Precision	$\frac{TP}{(TP+FP)}$
3	Recall	$\frac{TP}{(TP+FN)}$
4	F1 score	$2\ X\left\{\frac{(Precision\ X\ Accuracy)}{(Precision+\ Accuracy)}\right\}$
5	Sensitivity	$\frac{TP}{(TP+FN)}$
6	Specificity	$\frac{TN}{(TN+FP)}$
$Where,$ TP-True Positive, TN-True Negative, FP-False Positive, FN-False Negative		

3.5.1. Mirai (Botnet) dataset

As an initial step, data are pre-processed using three wavelet transforms: DWT, SWT, and TQWT. DWT is good in terms of representation of temporal changes of data, and SWT is efficients in terms of translation invariance and multi-resolution analysis. TQWT is better than DWT and SWT, resulting in features that are more relevant by predicting complex patterns in network traffic for further analysis. Statistical features are calculated from wavelet coefficients.

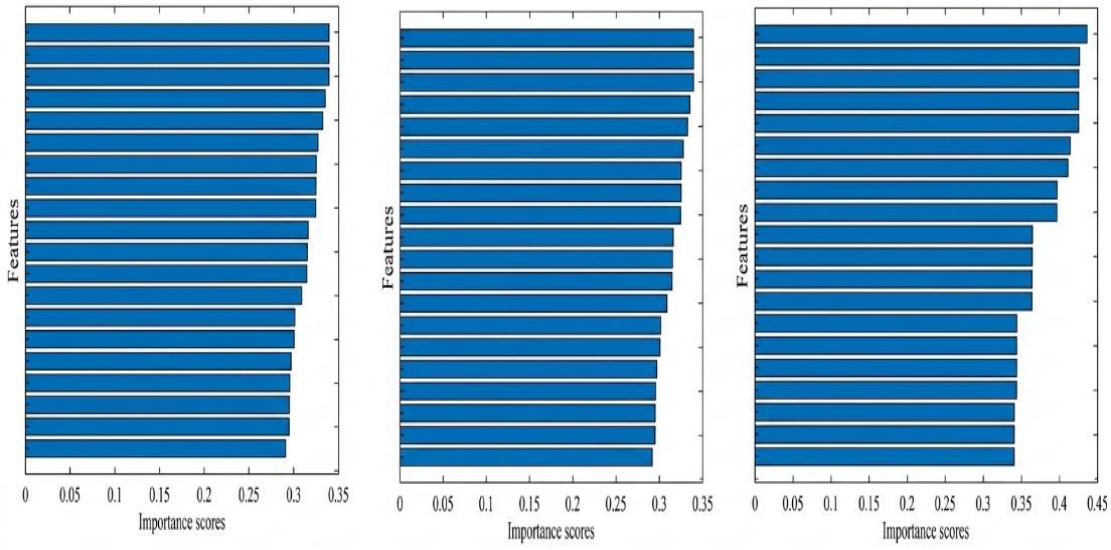
The distribution of energy features across bands for DWT, SWT, and TQWT are as shown in Figure 9. As datasets are of high dimension, the computation complexity of the algorithm also increases. Therefore, from wavelet features, highly significant features are chosen by computing correlation or similarity with the use of Pearson's Co-efficient of correlation. Highly significant features corresponding to different variables in the original data are chosen from DWT, SWT, and TQWT features using Pearson's coefficient of correlation, as shown in Figure 10. Features selected are fed as input to classifiers, cubic SVM, and cubic KNN to differentiate benign and malicious behaviour under a 70:30 ratio during the training and testing phase. The original dataset is of very high dimension; three datasets of size 10000, 20000, and 30000 samples are created to examine the performance of the proposed work. Figure 11 displays the scatter plot of the proposed methods. Confusion matrices obtained for Cubic KNN and Cubic SVM for different wavelet transform inputs are used to find various performance measures. Table 7 displays the performance metrics evaluated for the Mirai (Botnet) dataset of 10000 samples. Among three pre-processed data sets using DWT, SWT, and TQWT, the cubic SVM classifier performs superior compared to cubic KNN. From the results, TQWT based classifier result is good compared to DWT and SWT. Maximum metric value of 0.992 is achieved for TQWT-based cubic SVM (TCSVM). For TCKNN, metric value of 0.981 achieved for accuracy, 0.983 for recall, 0.984 for sensitivity, 0.982 for specificity. Comparing DWT and SWT results for the Mirai dataset of 10000 samples, SWT performs well. TQWT and SWT-based classifier results are better than DWT-based classifier results as both TQWT and SWT predicts varying features better than DWT, which suffers in loss of features due to down-sampling.



(a) DWT (b) SWT (c) TQWT

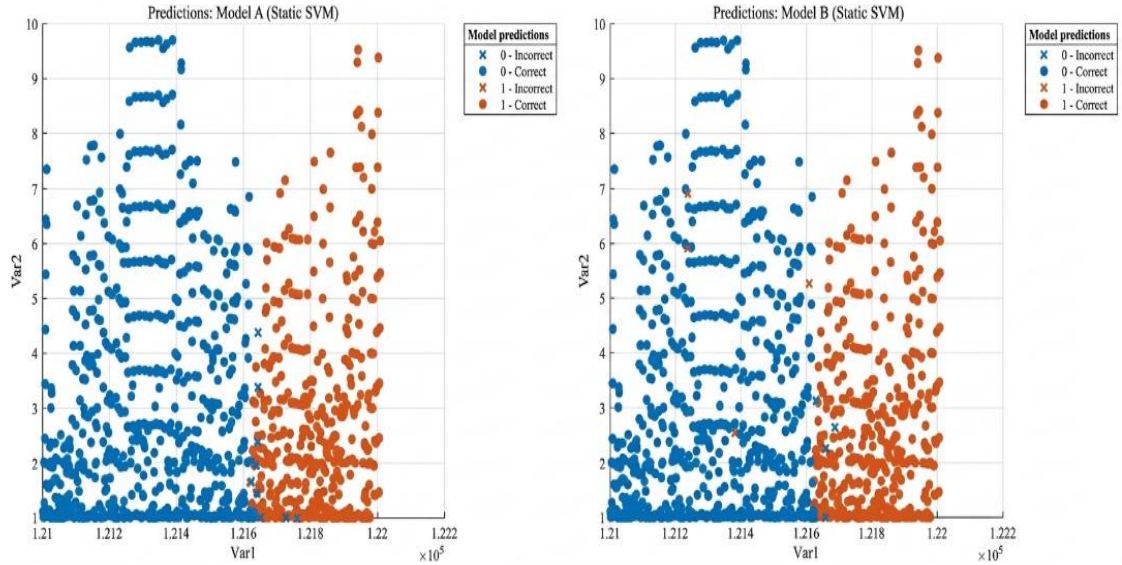
Fig. 9 Features extracted using wavelet transforms.

Figure 12 shows the comparison of all the proposed methods. TQWT achieved high performance metrics, indicating the detection of most attacks with maximum identification of positive cases and minimum false alarms.



(a) DWT (b) SWT (c) TQWT

Fig. 10 Features selected using Pearson's coefficient of correlation



(a) TCKNN (b) TCSVM

Fig. 11 Classification: Scatter plot results-Mirai (10000 samples)-TCKNN and TCSVM

Table 7. Mirai- Performance measures (10000 samples)

Mirai-10000 Samples						
Methods	Accuracy	Precision	Recall	F1 score	Sensitivity	Specificity
DCKNN	0.969	0.968	0.964	0.964	0.969	0.97
DCSVM	0.975	0.978	0.979	0.974	0.975	0.979
SCKNN	0.97	0.975	0.97	0.969	0.969	0.97
SCSVM	0.987	0.983	0.987	0.984	0.986	0.98
TCKNN	0.981	0.983	0.984	0.98	0.982	0.982
TCSVM	0.992	0.993	0.991	0.995	0.994	0.996

Table 8 displays the performance metrics evaluated for the Mirai dataset of 20000 samples. Maximum metric value of 0.993 is achieved for the proposed methods TCSVM, followed by TCKNN, SCSVM and SCSVM.

Figure 13 shows the comparison of all the proposed methods for various parameters. TQWT detects positive cases and minimum false alarms during an attack.

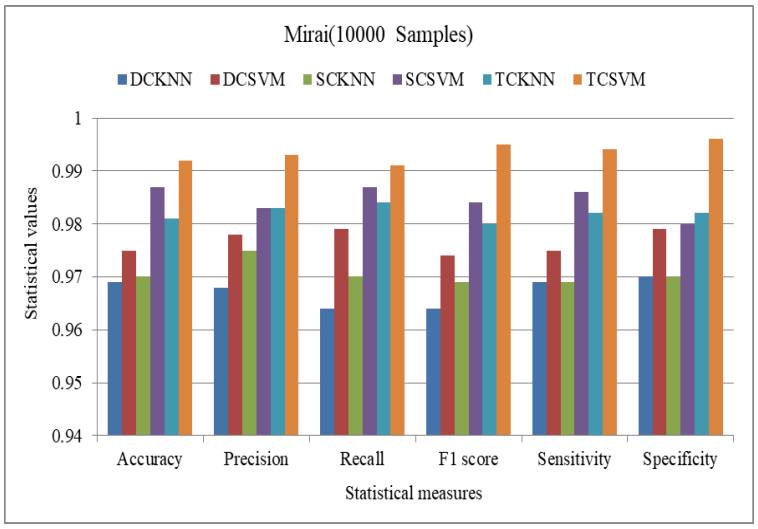


Fig. 12 Mirai-Performance comparison of proposed methods for various metrics (DCKNN, DCSVM, SCKNN, SCSVM, TCKNN, and TCSVM-10000 samples)

Table 8. Mirai- Performance measures (20000 samples)

Mirai-20000 Samples						
Methods	**Accuracy**	**Precision**	**Recall**	**F1 score**	**Sensitivity**	**Specificity**
DCKNN	0.969	0.969	0.966	0.965	0.969	0.972
DCSVM	0.977	0.978	0.978	0.976	0.976	0.978
SCKNN	0.972	0.977	0.972	0.971	0.972	0.972
SCSVM	0.988	0.985	0.988	0.986	0.987	0.982
TCKNN	0.982	0.984	0.986	0.982	0.983	0.983
TCSVM	0.993	0.995	0.993	0.996	0.995	0.997

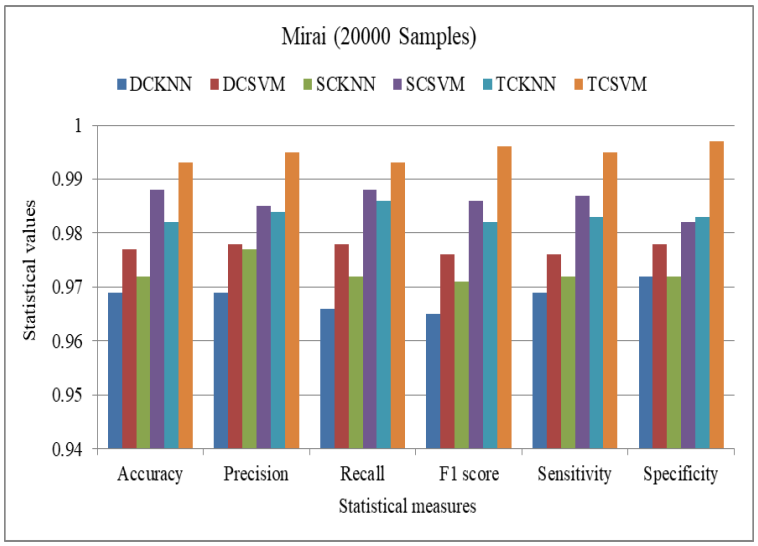


Fig. 13 Mirai-Performance comparison of proposed methods for various metrics (DCKNN, DCSVM, SCKNN, SCSVM, TCKNN, and TCSVM-20000 samples)

Table 9 displays the performance metrics evaluated for Mirai dataset of 30000 samples. As the samples taken into consideration increases, most of the proposed methods results in a maximum value for all the performance metrics. Maximum metric value achieved for all the parameters for the proposed method TCSVM. Among cubic KNN methods, TCKNN results are superior in comparison to DCKNN and SCKNN. Similarly, among cubic SVM methods, TCSVM performs better compared to DCSVM and SCSVM.

Figure 14 shows the comparison of all the proposed methods. TQWT achieved high performance metrics during

attack detection. The proposed algorithms compared with existing methods. To validate this, the Mirai dataset of 30000 samples is taken. The traditional algorithm is compared with the proposed kernel-based classifiers. In addition, these comparison results validate the proposed Wavelet transform-based preprocessing in comparison to PCA preprocessed classifiers for detecting benign and malicious behaviors in network traffic. Cubic SVM classifier methods DCSVM, SCSVM and TCSVM performance is superior in terms of all the metrics compared to all the other methods, as shown in Table 10. Performance of PKNN and PSVM is better than LR but lesser than the proposed methods.

Table 9. Mirai- Performance measures (30000 samples)

Mirai-30000 Samples						
Methods	**Accuracy**	**Precision**	**Recall**	**F1 score**	**Sensitivity**	**Specificity**
DCKNN	0.971	0.973	0.971	0.969	0.971	0.974
DCSVM	0.979	0.981	0.982	0.979	0.979	0.978
SCKNN	0.975	0.978	0.973	0.973	0.973	0.975
SCSVM	0.989	0.987	0.989	0.987	0.988	0.984
TCKNN	0.983	0.984	0.987	0.983	0.984	0.984
TCSVM	0.995	0.996	0.995	0.997	0.996	0.997

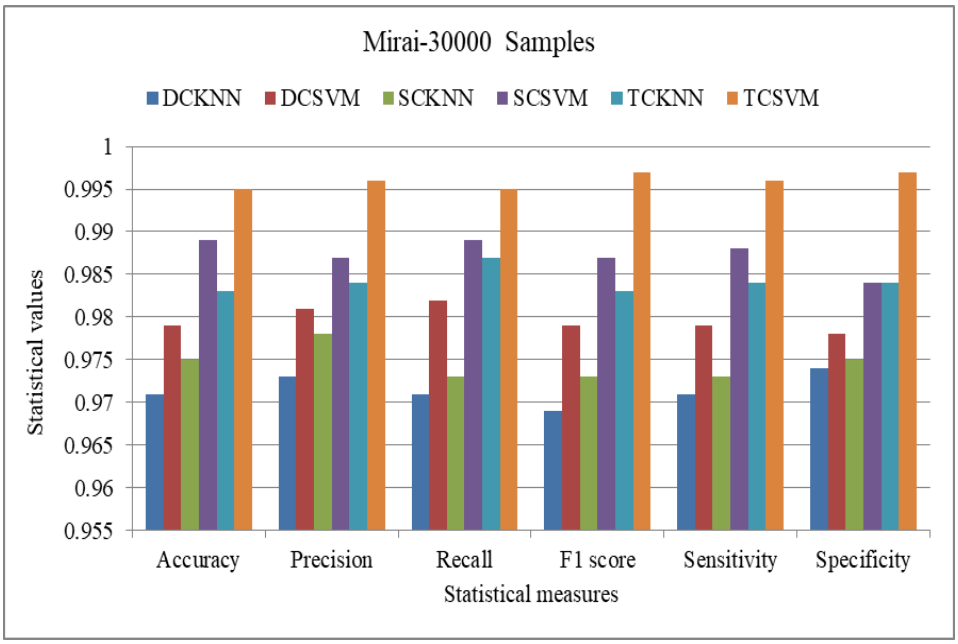


Fig. 14 MIRAI-Performance comparison of proposed methods for various metrics (DCKNN, DCSVM, SCKNN, SCSVM, TCKNN, and TCSVM-30000 samples)

Figures 15, 16 and 17 display the comparison of proposed methods with existing LR, PKNN and PSVM methods, which perform better than existing.

Table 10. Mirai- Performance comparison with existing methods

Mirai- comparison with existing methods						
Methods	**Accuracy**	**Precision**	**Recall**	**F1 score**	**Sensitivity**	**Specificity**
LR [10]	0.865	0.696	0.92	0.72	0.91	0.89
PKNN [18]	0.952	0.954	0.953	0.958	0.952	0.959
PSVM [28]	0.961	0.959	0.964	0.954	0.961	0.962
DCKNN	0.971	0.973	0.971	0.969	0.971	0.974
DCSVM	0.979	0.981	0.982	0.979	0.979	0.978
SCKNN	0.975	0.978	0.973	0.973	0.973	0.975
SCSVM	0.989	0.987	0.989	0.987	0.988	0.984
TCKNN	0.983	0.984	0.987	0.983	0.984	0.984
TCSVM	0.995	0.996	0.995	0.997	0.996	0.997

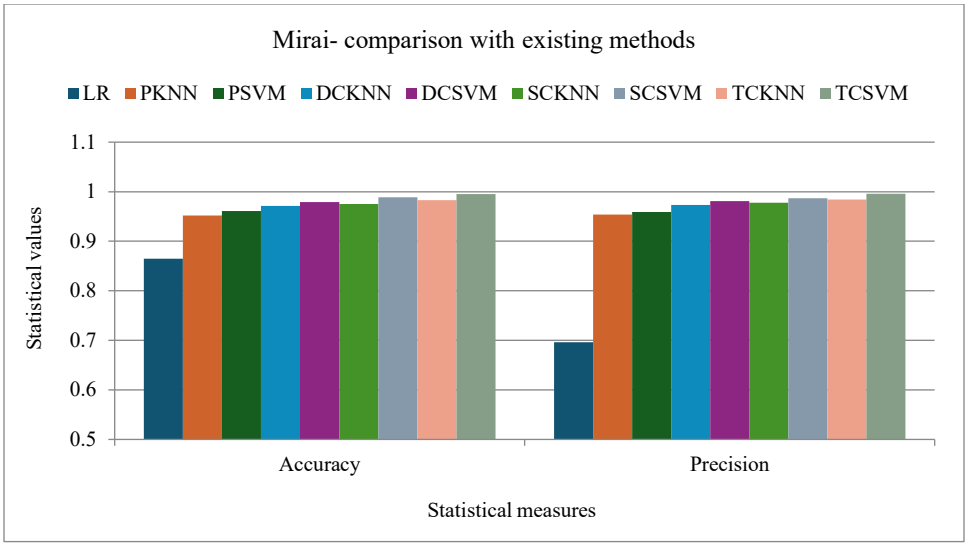


Fig. 15 Comparison of proposed methods (Accuracy and Precision)

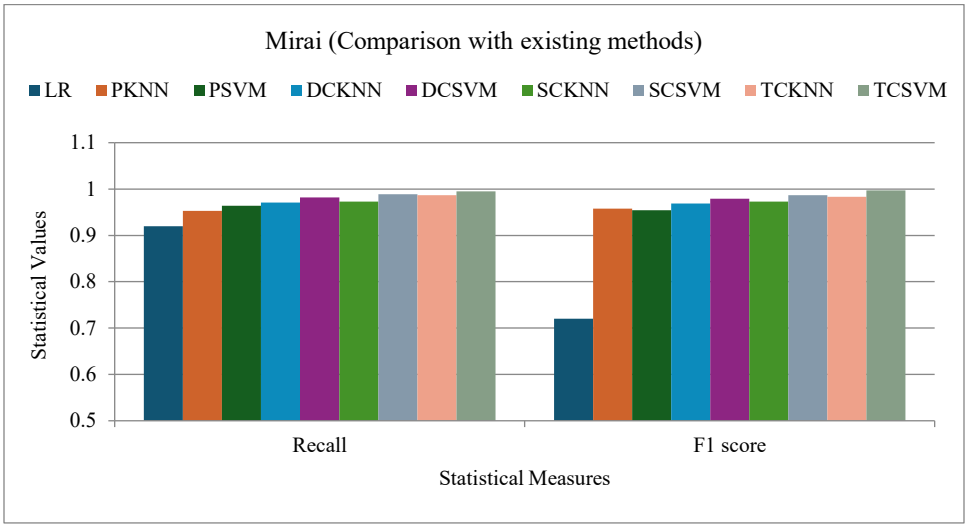


Fig. 16 Comparison of proposed methods (Recall and F1 Score)

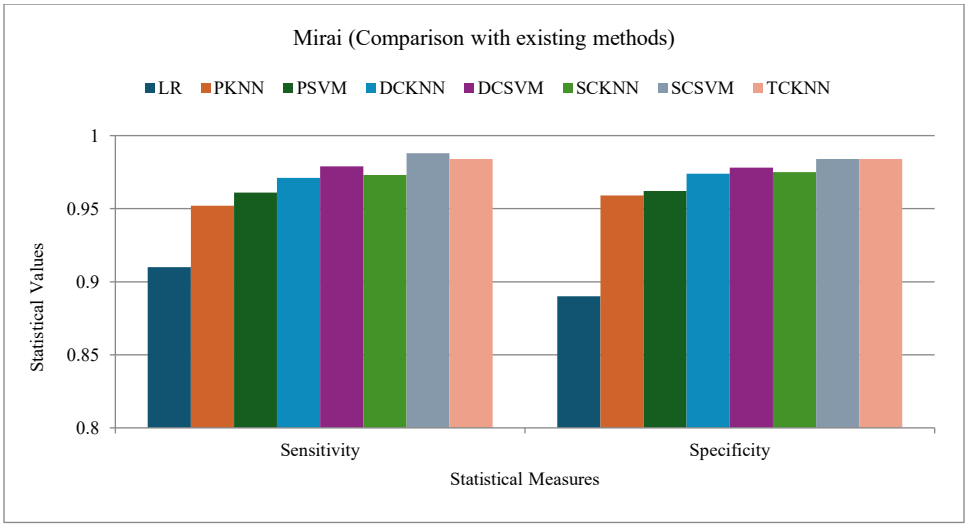


Fig. 17 Performance comparison of proposed methods with existing methods (Sensitivity and Specificity)

3.5.2. DoS dataset

The second dataset used to test the proposed methods is DoS dataset. Table 11 displays the performance metrics evaluated for Dos dataset of 10000 samples. Accuracy value of 0.990 and precision value of 0.989 is achieved by proposed method TCSVM.

Maximum recall and F1 score obtained for proposed method TCSVM is 0.991 and 0.988 respectively. Similarly, maximum sensitivity and specificity achieved for TCSVM. Figure 18 shows the comparison of all the proposed methods. TCSVM prediction minimizes false positives more than TQWT features, than DWT and SWT features.

Table 11. DoS- Performance measures (10000 samples)

DoS-10000 Samples						
Methods	**Accuracy**	**Precision**	**Recall**	**F1 score**	**Sensitivity**	**Specificity**
DCKNN	0.972	0.975	0.973	0.972	0.974	0.976
DCSVM	0.975	0.976	0.974	0.974	0.975	0.978
SCKNN	0.979	0.981	0.981	0.984	0.979	0.984
SCSVM	0.981	0.983	0.984	0.985	0.981	0.986
TCKNN	0.989	0.986	0.989	0.984	0.987	0.99
TCSVM	0.990	0.989	0.991	0.988	0.990	0.992

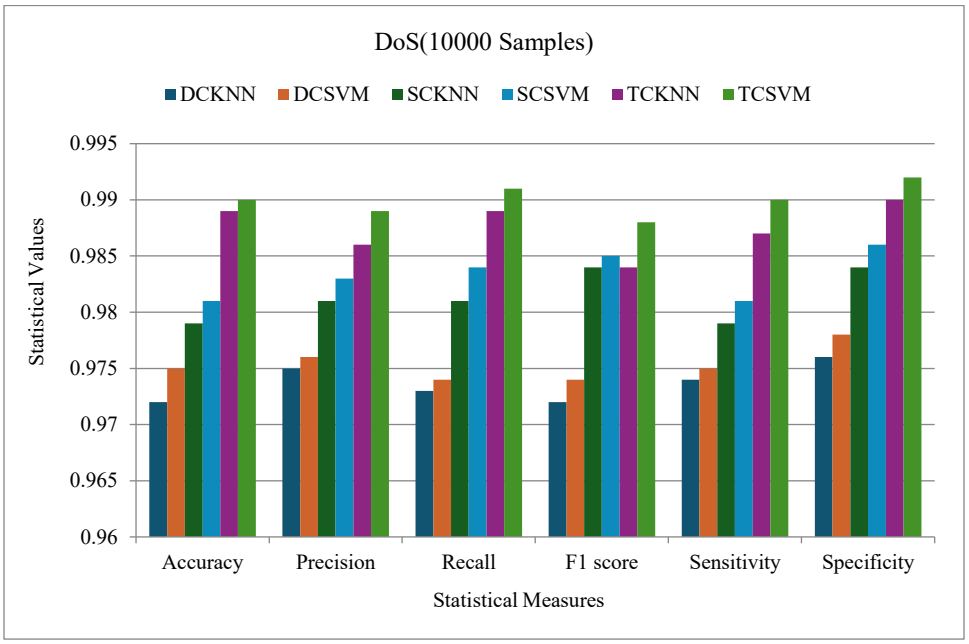


Fig. 18 DoS-Performance comparison of proposed methods for various metrics (DCKNN, DCSVM, SCKNN, SCSVM, TCKNN, and TCSVM-10000 samples)

Table 12 displays the performance metrics evaluated for DoS dataset of 20000 samples. Maximum metric values achieved for proposed TQWT based SVM classifier TCSVM followed by TCKNN, SCSVM, SCKNN, DCSVM and DCKNN. Accuracy value of 0.975 for DCSVM, 0.983 for SCSVM and 0.992 for TCSVM obtained for proposed SVM methods followed by 0.973 for DCKNN, 0.981 for SCKNN and 0.988 for TCKNN obtained for proposed KNN respectively. Figure 19 shows the comparison of all the proposed methods.

Table 12. DoS- Performance measures (20000 samples)

DoS-20000 Samples						
Methods	**Accuracy**	**Precision**	**Recall**	**F1 score**	**Sensitivity**	**Specificity**
DCKNN	0.973	0.975	0.974	0.973	0.975	0.977
DCSVM	0.975	0.976	0.973	0.975	0.976	0.978
SCKNN	0.981	0.983	0.982	0.984	0.979	0.985
SCSVM	0.983	0.984	0.985	0.986	0.983	0.986
TCKNN	0.988	0.987	0.988	0.985	0.987	0.989
TCSVM	0.992	0.992	0.992	0.991	0.992	0.993

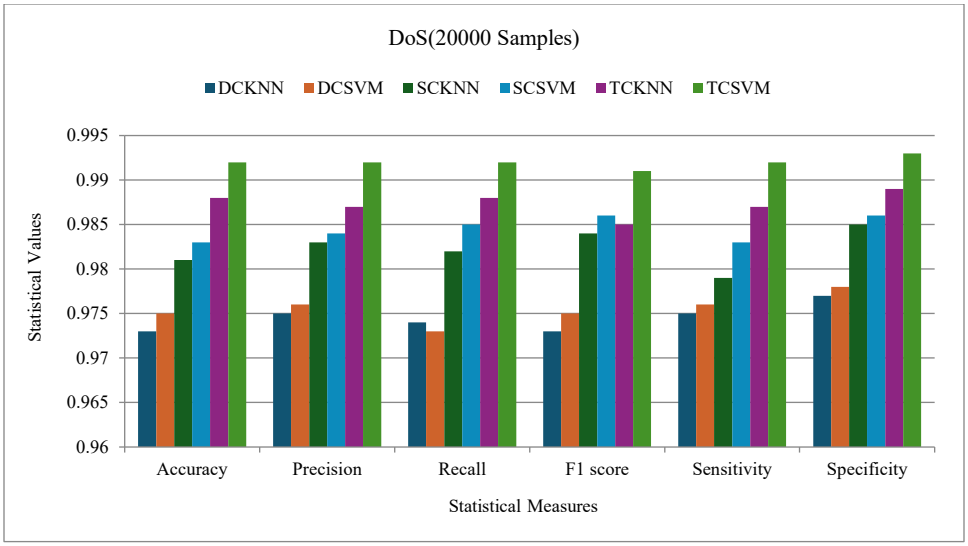


Fig. 19 DoS-Performance comparison of proposed methods for various metrics (DCKNN, DCSVM, SCKNN, SCSVM, TCKNN, and TCSVM-20000 samples)

Table 13 displays the performance metrics evaluated for the DoS dataset of 30000 samples, which contains more non-linearities and is an example for complex dataset. Maximum accuracy and precision value of 0.995 obtained for TCSVM for this complex DoS dataset. Among the cubic KNN proposed methods, TCKNN performance is superior compared to DCKNN and SCKNN. Figure 20 shows the comparison of all the proposed methods for various parameters: accuracy, precision, recall, F1 score, sensitivity and specificity.

Table 13. DoS- Performance measures (30000 samples)

DoS-30000 Samples						
Methods	**Accuracy**	**Precision**	**Recall**	**F1 score**	**Sensitivity**	**Specificity**
DCKNN	0.975	0.977	0.976	0.976	0.977	0.978
DCSVM	0.977	0.978	0.975	0.977	0.978	0.979
SCKNN	0.982	0.984	0.983	0.986	0.981	0.986
SCSVM	0.985	0.986	0.985	0.988	0.985	0.988
TCKNN	0.992	0.989	0.991	0.989	0.992	0.993
TCSVM	0.995	0.995	0.995	0.993	0.994	0.995

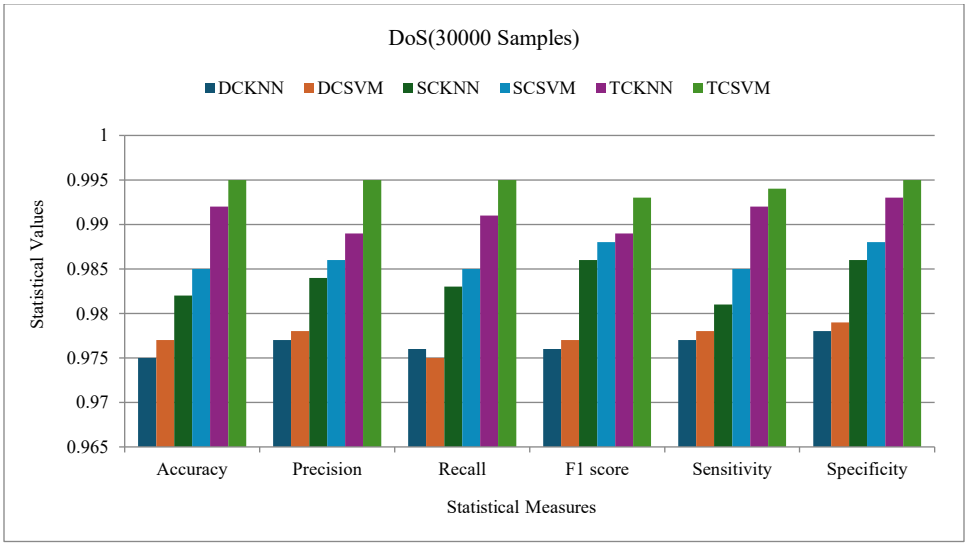


Fig. 20 Performance comparison of proposed methods for various metrics (DCKNN, DCSVM, SCKNN, SCSVM, TCKNN, and TCSVM-30000 samples)

To validate this, the DoS dataset of 30000 samples is taken. Performance of the Wavelet preprocessed kernel-based classifier results is good compared to LR and PCA-based traditional classifier results. Cubic SVM and Cubic KNN produced comparable results and were better than existing methods. Table 14, Figure 21, Figure 22 and Figure 23 display the accuracy, precision, recall, F1 score, sensitivity and specificity comparison of proposed methods with existing LR., PKNN and PSVM methods. High value obtained for TCSVM followed by TCKNN.

Table 14. DoS- Performance comparison with existing methods

DoS- Performance comparison with existing methods						
Methods	**Accuracy**	**Precision**	**Recall**	**F1 score**	**Sensitivity**	**Specificity**
LR	0.86	0.59	0.91	0.72	0.91	0.88
PKNN	0.961	0.967	0.962	0.966	0.964	0.966
PSVM	0.964	0.969	0.964	0.969	0.967	0.969
DCKNN	0.975	0.977	0.976	0.976	0.977	0.978
DCSVM	0.977	0.978	0.975	0.977	0.978	0.979
SCKNN	0.982	0.984	0.983	0.986	0.981	0.986
SCSVM	0.985	0.986	0.985	0.988	0.985	0.988
TCKNN	0.992	0.989	0.991	0.989	0.992	0.993
TCSVM	0.995	0.995	0.995	0.993	0.995	0.995

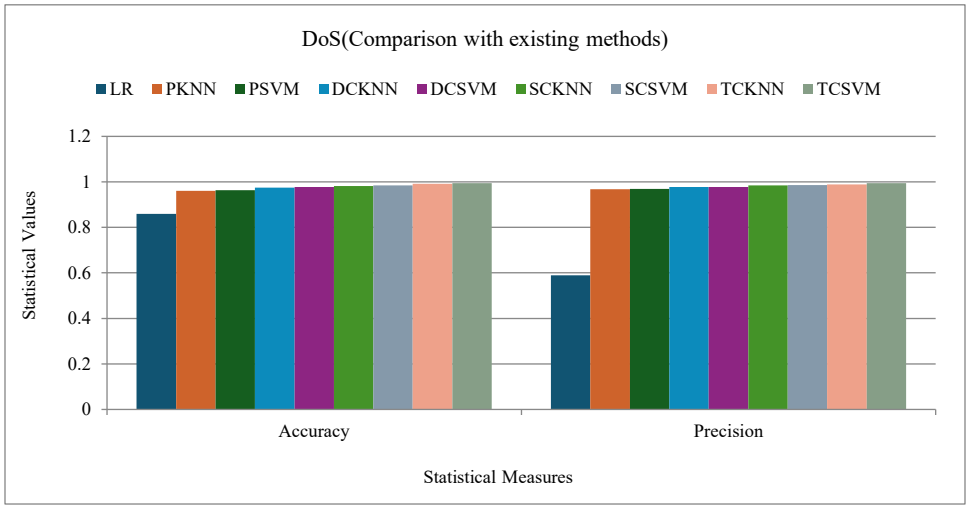


Fig. 21 Performance comparison of proposed methods with existing methods (Accuracy and Precision)

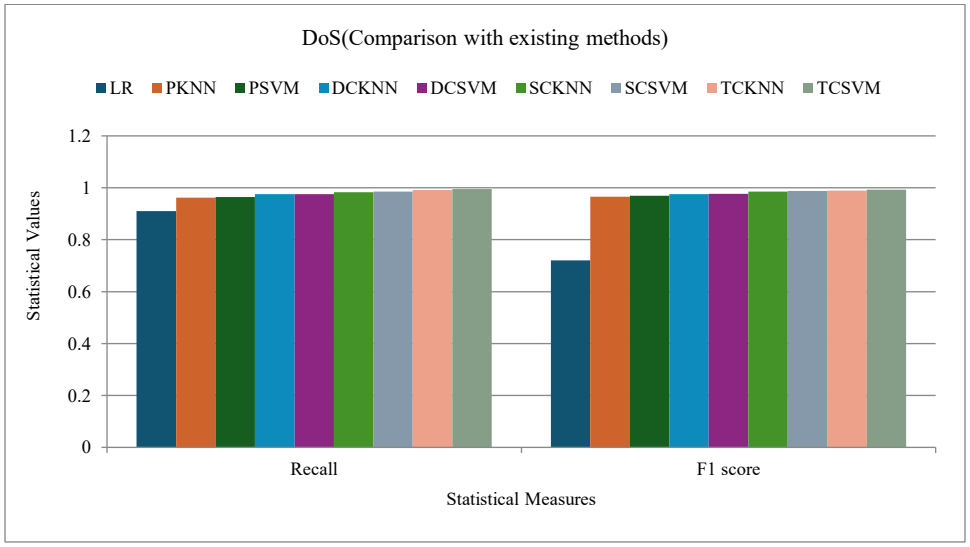


Fig. 22 Performance comparison of proposed methods with existing methods (Recall and Precision)

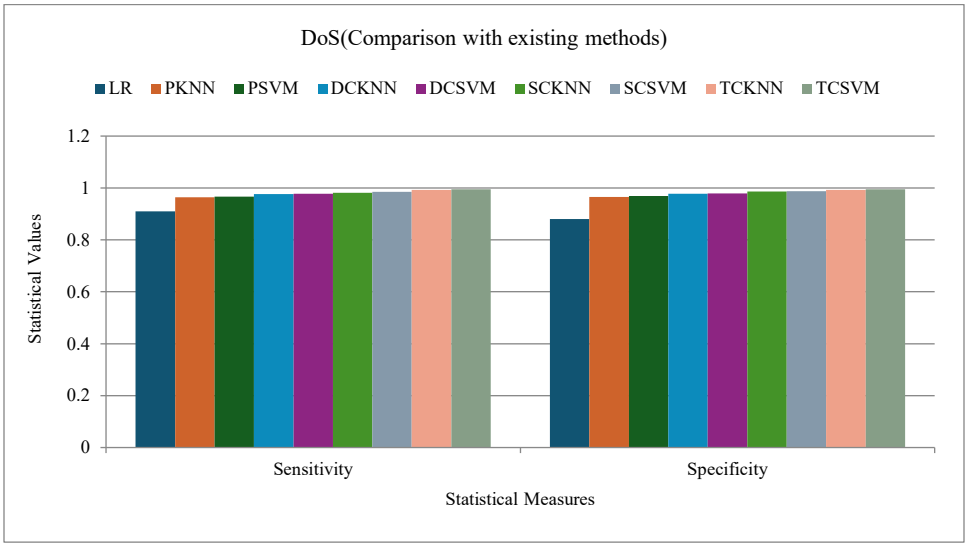


Fig. 23 Performance comparison of proposed methods with existing methods (Sensitivity and Specificity)

3.5.3. MiM dataset

Third dataset utilized to test the proposed method is the Man in the Middle (MiM) dataset. Table 15 displays the performance metrics evaluated for the MiM dataset of 10000 samples. The maximum performance metric value is achieved for TQWT based SVM classifier compared to other proposed methods. Figure 24 shows the comparison of all the proposed methods. The maximum value of accuracy, recall, and sensitivity obtained for the proposed TCSVM method is 0. 994.TCSVM is better compared to TCKNN. Similarly, SCSVM is better than SCKNN, and DCSVM is better compared to DCKNN.

Table 15. MiM-Performance measures (10000 samples)

MiM-10000 Samples						
Methods	**Accuracy**	**Precision**	**Recall**	**F1 score**	**Sensitivity**	**Specificity**
DCKNN	0.982	0.979	0.984	0.983	0.983	0.985
DCSVM	0.985	0.981	0.986	0.984	0.984	0.986
SCKNN	0.988	0.982	0.988	0.986	0.985	0.988
SCSVM	0.991	0.988	0.992	0.989	0.989	0.991
TCKNN	0.993	0.992	0.993	0.991	0.993	0.994
TCSVM	0.994	0.993	0.994	0.992	0.994	0.996

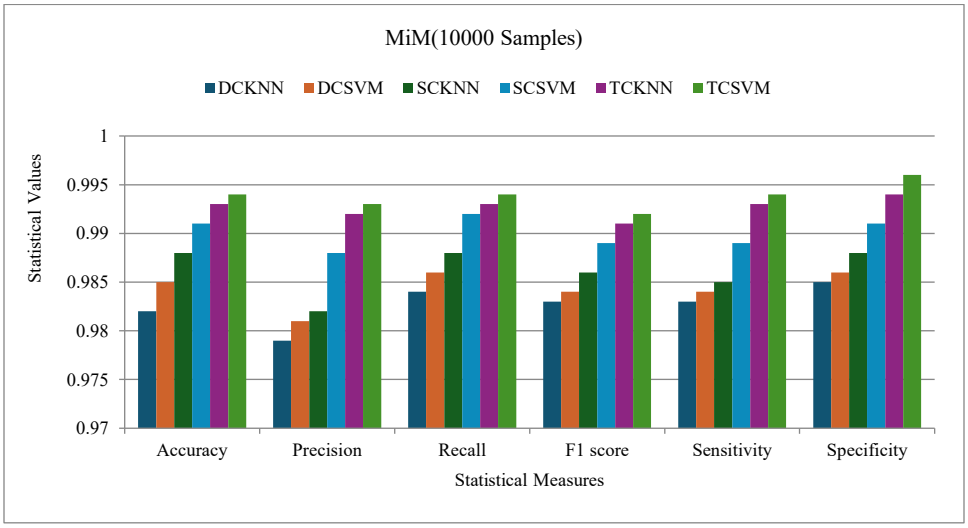


Fig. 24 MiM-Performance comparison of proposed methods for various metrics (DCKNN, DCSVM, SCKNN, SCSVM, TCKNN, and TCSVM-10000 samples)

Table 16 displays the performance metrics evaluated for MiM dataset of 20000 samples. TQWT achieved an accuracy of 0.994 and 0.997 for cubic SVM and cubic KNN classifiers compared to DWT and SWT based classifier results.

Figure 25 shows the comparison of all the proposed methods for various parameters accuracy, precision, recall, F1 score, sensitivity and specificity.

Table 16. MiM-Performance measures (20000 samples)

MiM-20000 Samples						
Methods	**Accuracy**	**Precision**	**Recall**	**F1 score**	**Sensitivity**	**Specificity**
DCKNN	0.983	0.979	0.985	0.984	0.985	0.986
DCSVM	0.986	0.982	0.987	0.985	0.985	0.986
SCKNN	0.988	0.983	0.988	0.987	0.986	0.989
SCSVM	0.992	0.989	0.994	0.991	0.989	0.992
TCKNN	0.994	0.993	0.994	0.992	0.994	0.993
TCSVM	0.997	0.994	0.996	0.994	0.995	0.997

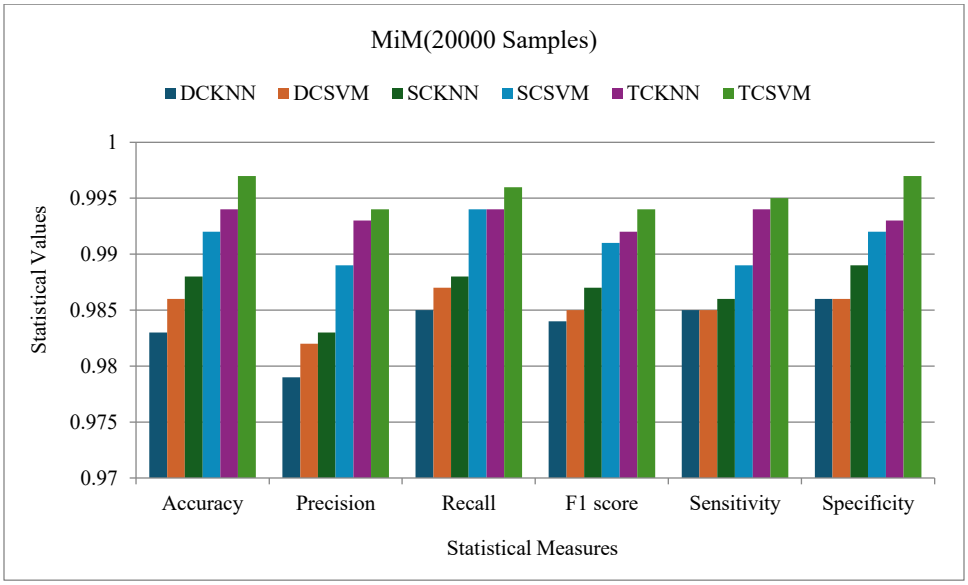


Fig. 25 MiM Dataset-Performance comparison of proposed methods for various metrics (DCKNN, DCSVM, SCKNN, SCSVM, TCKNN, and TCSVM-20000 samples)

Table 17 displays the performance metrics evaluated for MiM dataset of 30000 samples. TQWT achieved an accuracy of 0.995 and 0.997 for cubic SVM and cubic KNN classifiers compared to DWT and SWT based classifier results.

Figure 26 shows the comparison of all the proposed methods for various parameters accuracy, precision, recall, F1 score, sensitivity and specificity.

Table 17. MiM-Performance measures (30000 samples)

MiM-10000 Samples						
Methods	**Accuracy**	**Precision**	**Recall**	**F1 score**	**Sensitivity**	**Specificity**
DCKNN	0.984	0.981	0.986	0.984	0.985	0.986
DCSVM	0.987	0.984	0.988	0.987	0.986	0.987
SCKNN	0.988	0.984	0.989	0.988	0.987	0.989
SCSVM	0.993	0.989	0.995	0.993	0.991	0.993
TCKNN	0.995	0.994	0.996	0.994	0.994	0.995
TCSVM	0.997	0.995	0.998	0.996	0.996	0.998

The performance of proposed algorithms is validated in comparison with existing methods namely LR, PCA preprocessed KNN and SVM classifiers denoted as PKNN and PSVM. To validate this, MiM dataset of 30000 samples are taken. Maximum performance metric value achieved for all the parameters for proposed cubic SVM based methods DCSVM, SCSVM and TCSVM.

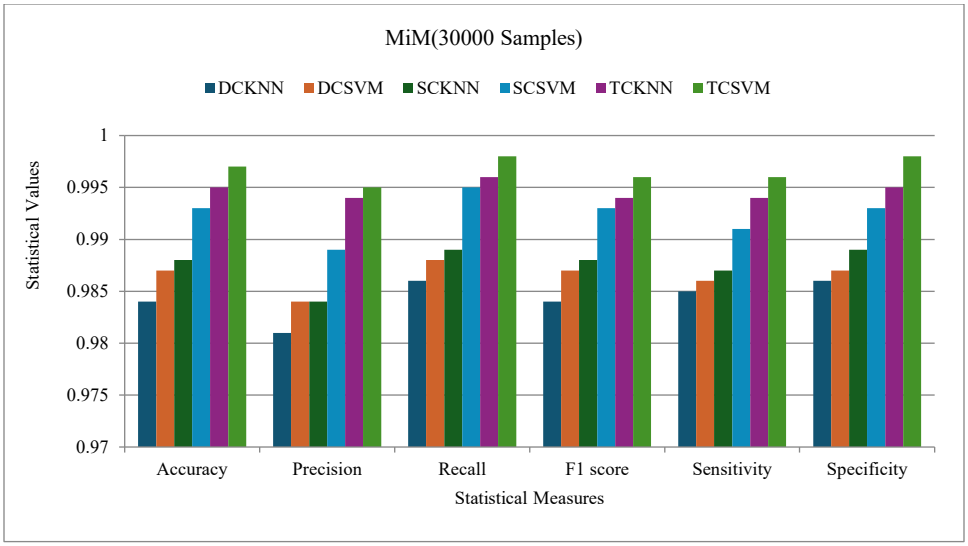


Fig. 26 MiM-Performance comparison of proposed methods for various metrics (DCKNN, DCSVM, SCKNN, SCSVM, TCKNN, and TCSVM-30000 samples)

Cubic kernel-based methods produced better results than existing methods. Table 18, Figure 27, Figure 28 and Figure 29 display the comparison of proposed methods with existing LR. PKNN and PSVM methods.

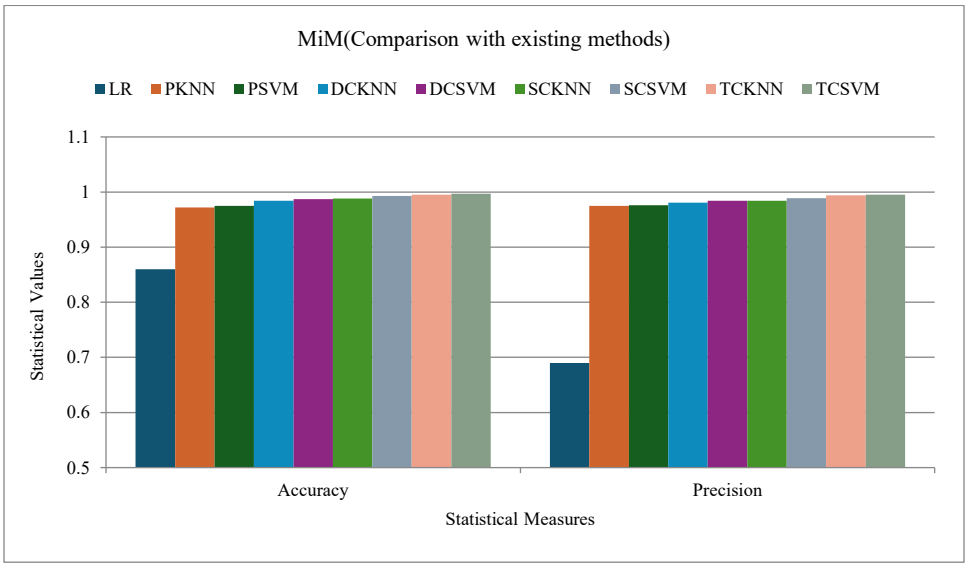


Fig. 27 Performance comparison of proposed methods with existing methods (Accuracy and Precision)

Table 18. MiM-Performance comparison with existing methods

MiM-Performance comparison with existing methods						
Methods	**Accuracy**	**Precision**	**Recall**	**F1 score**	**Sensitivity**	**Specificity**
LR [10]	0.86	0.69	0.91	0.72	0.92	0.91
PKNN [28]	0.972	0.975	0.976	0.972	0.973	0.977
PSVM [18]	0.975	0.976	0.977	0.974	0.975	0.979
DCKNN	0.984	0.981	0.986	0.984	0.985	0.986
DCSVM	0.987	0.984	0.988	0.987	0.986	0.987
SCKNN	0.988	0.984	0.989	0.988	0.987	0.989
SCSVM	0.993	0.989	0.995	0.993	0.991	0.993
TCKNN	0.995	0.994	0.996	0.994	0.994	0.995
TCSVM	0.997	0.995	0.998	0.996	0.996	0.998

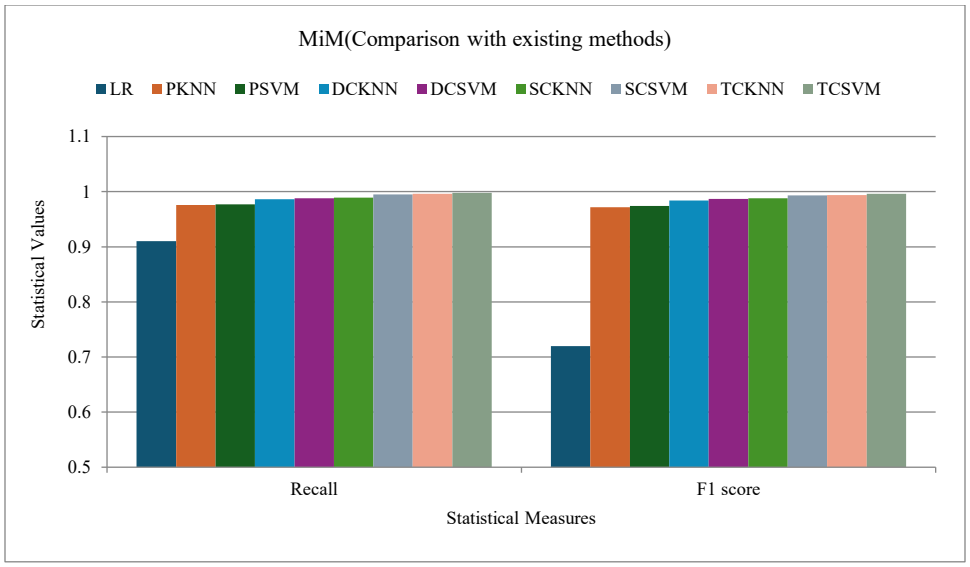


Fig. 28 Performance comparison of proposed methods with existing methods (Recall and F1 Score)

In summary, TQWT based SVM results obtained high performance metric values in comparison with Cubic KNN method.

Similarly, SWT and DWT based SVM results are good compared to Cubic KNN classifier. Cubic KNN results are better than existing LR, PSVM and PKNN methods.

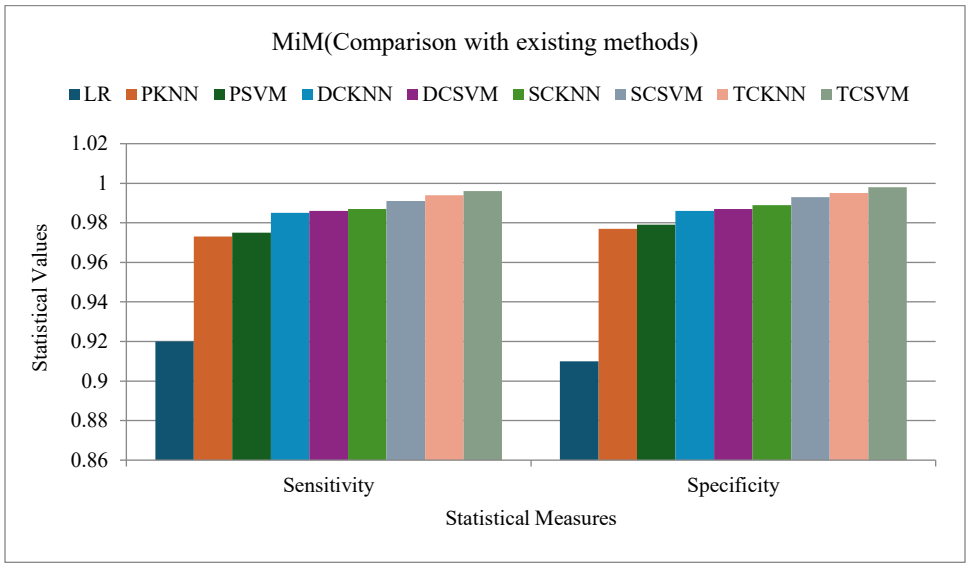


Fig. 29 Performance comparison of proposed methods with existing methods - Sensitivity and Specificity

3.5.4. Discussion

The proposed work is analyzed on network traffic created in video surveillance or IoT networks. The methodology is tested for the classification of benign and malicious behaviour in three different types of attacks, namely Botnet, DoS, and Mirai. Maximum accuracy of 99.7% achieved for the classification with MiM attacks, followed by an average accuracy value of 99.5% in Botnet and DoS attacks. Similarly, other performance measures F1 score for MiM, followed by DoS and Mirai indicating that the proposed feature extraction-based classifiers predict high actual positive instances and minimum false positives. Dataset contain maximum of 115 unknown features. Pre-processing using DWT, SWT, and TQWT, followed by feature extraction, reduces 115 features to 20 features for further processing. In addition, cubic kernel function-based classifiers improve the accuracy by expanding the feature set in the hyperplane in the Cubic SVM classifier and by assigning different weighting factors in KNN. In terms of computation analysis, feature extraction and only inner product operation in Cubic SVM minimizes the computational complexity of the proposed SVM-based methods. Similarly, Cubic KNN on pre-processed data reduces computational complexity compared to traditional KNN methods.

3.5.5. Ablation Study

The feature extraction stage plays a vital role in the proposed methodology in increasing the accuracy measure and reducing the computational complexity of the algorithm in the classification of benign and malicious behaviour in network traffic created on video surveillance networks. This is validated with the findings of the proposed work on the Mirai (Botnet) dataset. Table 19 shows the metric values for the Mirai dataset (30000 samples) with and without a pre-processing stage, followed by Cubic SVM and Cubic KNN classifiers. From both Cubic KNN and Cubic SVM results with and without pre-processed data, an accuracy of 96.8 and 97.1% is achieved with all 115 features (without pre-processing stage), and an accuracy value of 97.9% is achieved with DWT,98.9% with SWT, and 99.5% with TQWT is obtained with only 20 features processed further to the Cubic SVM classifier. The proposed wavelet transform-based feature selection results are superior in terms of accuracy, with 20 reduced features from the original 115 features in the Mirai network traffic dataset. Cubic SVM results are superior in overall performance compared to cubic Kernel and without pre-processing based classifier methods.

Table 19. Ablation study

Methodology	Accuracy		Precision		Recall		F1 Score	
	Cubic KNN	Cubic SVM	Cubic KNN	Cubic SVM	Cubic KNN	Cubic SVM	Cubic KNN	Cubic SVM
Without pre-processing (115 features) (all features are used)	0.968	0.971	0.964	0.973	0.962	0.97	0.967	0.974
With DWT Pre-processed data (20 features) (Significant features and suffers due to down-sampling)	0.971	0.979	0.973	0.981	0.971	0.982	0.969	0.979
With SWT Pre-processed data (20 features) (better than DWT since it processes data of original size and fails to predict complex network behavior)	0.975	0.989	0.978	0.987	0.973	0.989	0.973	0.987
With TQWT Pre-processed data (20 features) (predicts complex data patterns in network traffic through non-uniform frequency partitioning)	0.983	0.995	0.984	0.996	0.987	0.995	0.983	0.997

4. Conclusion

The proposed method combines wavelet transform-based feature extraction with kernel classifiers to classify behaviour as benign or malicious in three types of attacks: Botnet, DoS, and MiM. The methodology is evaluated using Kitsune datasets, which are generated by transmitting packets in video surveillance and IoT-populated networks to simulate normal network traffic. The approach selects features from DWT, SWT, and TQWT, processing only 20 out of the original 115 unknown features to classify abnormal activities. Two classifiers, cubic SVM and Cubic KNN, are used to analyse the results across three different attack datasets. In the pre-processing stage, TQWT demonstrates superior performance, while in the post-processing step, the Cubic SVM classifier outperforms Cubic KNN in distinguishing benign and malicious behaviour in a network. In summary, the performance of the proposed TCSVM is better compared to other proposed and existing methods, with an accuracy value of 0.995 for both Botnet and DoS datasets and 0.997 for the MiM attack dataset. Future work will focus on testing this methodology on larger and more complex datasets involving various attacks and integrating additional machine learning models to enhance the classification of abnormal activities in IoT networks.

Conflicts of Interest

The author(s) declare(s) that there is no conflict of interest regarding the publication of this paper.

Funding Statement

The authors received no specific funding for this article.

Acknowledgments

S. Divya: Conceptualization, Methodology, Software, Writing - original draft. D. Kavitha: Supervision, Review, Editing, Validation.

Supplementary Material

https://github.com/divya2020research-bit/supplementary-file

References

[1] Ogugua Chimezie Obi et al., "Review of Evolving Cloud Computing Paradigms: Security, Efficiency, and Innovations," *Computer Science and IT Research Journal*, vol. 5, no. 2, pp. 270-292, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[2] Muhammad Dawood et al., "Cyberattacks and Security of Cloud Computing: A Complete Guideline," *Symmetry*, vol. 15, no. 11, pp. 1-33, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[3] Rajakumaran Gayathri et al., "Detection and Mitigation of IoT-based Attacks using SNMP and Moving Target Defense Techniques," *Sensors*, vol. 23, no. 3, pp. 1-13, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[4] Hamidreza Fereidouni, Olga Fadeitcheva, and Mehdi Zalai "IoT and Man-in-the-Middle Attacks," *Security and Privacy*, vol. 8, no. 2, pp. 1-19, 2025. [CrossRef] [Google Scholar] [Publisher Link]

[5] Shefali Madan, Anita, and Ashif Ali, "DDos Attacks in Cloud Environment," *International Journal of Health Sciences*, vol. 6, no. S4, pp. 5836-5847, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[6] Ziyad R. Alashhab et al., "Distributed Denial of Service Attacks Against Cloud Computing Environment: Survey, Issues, Challenges and Coherent Taxonomy," *Applied Sciences*, vol. 12, no. 23, pp. 1-32, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[7] R.S. Aashmi, and T. Jaya, "Detecting and Preventing of Attacks in Cloud Computing using Hybrid Algorithm," *Intelligent Automation and Soft Computing*, vol. 35, no. 1, pp. 79-95, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[8] Ravindra Kumar Chouhan, Mithilesh Atulkar, and Naresh Kumar Nagwani "A Distributed Attack Detection System for SDN using Stack of Classifiers," *International Journal of Engineering Trends and Technology*, vol. 71, no. 3, pp. 81-90, 2023. [CrossRef] [Publisher Link]

[9] Ahmad Ahmad et al., "Tuning Data Preprocessing Techniques for Improved Wind Speed Prediction," *Energy Reports*, vol. 11, pp. 287-303, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[10] Hari Gonaygunta, "Machine Learning Algorithms for Detection of Cyber Threats using Logistic Regression," *International Journal of Smart Sensor and Adhoc Network*, vol. 3, no. 4, pp. 35-42, 2023. [CrossRef] [Google Scholar]

[11] Álvaro Michelena et al., "A Novel Intelligent Approach for Man-in-the-Middle Attacks Detection over Internet of Things Environments based on Message Queuing Telemetry Transport," *Expert Systems*, vol. 41, no. 2, pp. 1-15, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[12] Morgan Reece et al., "Defending Multi-Cloud Applications Against Man-in-the-Middle Attacks," *Proceedings of the 29th ACM Symposium on Access Control Models and Technologies*, Association for Computing Machinery, New York, NY, United States, pp. 47-52, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[13] Asha Varma Songa, and Ganesh Reddy Karriu, "An Integrated SDN Framework for early Detection of DDoS Attacks in Cloud Computing," *Journal of Cloud Computing*, vol. 13, no. 1, pp. 1-22, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[14] Mahdi Nsaif Jasim, and Methaq Talib Gaata, "K-Means Clustering-based Semi-Supervised for DDoS Attacks Classification," *Bulletin of Electrical Engineering and Informatics*, vol. 11, no. 6, pp. 3570-3576, 2022. [Google Scholar]

[15] Haya Alubaidan et al., "DDoS Detection in Software-Defined Network using Machine Learning," *International Journal of Cybernetics and Informatics*, vol. 12, no. 4, pp. 93-104, 2023. [Google Scholar]

[16] Lal Mohan Pattnaik et al., "Cloud DDoS Attack Detection Model with Data Fusion and Machine Learning Classifiers," *EAI Endorsed Transactions on Scalable Information Systems*, vol. 10, no. 6, pp. 1-9, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[17] Gottapu Sankara Rao, and P. Krishna Subbarao, "A Novel Framework for Detection of DoS/ DDoS Attack using Deep Learning Techniques," *International Journal of Intelligent Systems and Applications in Engineering*, vol. 12, no. 1, pp. 450-466, 2024. [Google Scholar] [Publisher Link]

[18] Furqan Rustam et al., "Denial of Service Attack Classification using Machine Learning with Multi-Features," *Electronics*, vol. 11, no. 22, pp. 1-30, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[19] Yisroel Mirsky et al., "Kitsune: An Ensemble of Autoencoders for Online Network Intrusion Detection," *arXiv*, pp. 1-15, 2018. [CrossRef] [Google Scholar] [Publisher Link]

[20] Ashley Woodiss-Field, Michael N. Johnstone, and Paul Haskell-Dowland, "Examination of Traditional Botnet Detection on IoT-based Bots," *Sensors*, vol. 24, no. 3, pp. 1-25, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[21] Shamshair Ali et al., "A Novel Approach of Botnet Detection using Hybrid Deep Learning for Enhancing Security in IoT Networks," *Alexandria Engineering Journal*, vol. 103, pp. 88-97, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[22] S.K. Khaja Shareef et al., "Enhanced Botnet Detection in IoT Networks using Zebra Optimization and Dual-Channel GAN Classification," *Scientific Reports*, vol. 14, no. 1, pp. 1-19, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[23] Swapna Thota, and D. Menaka, "Botnet Detection in the Internet-of-Things Networks using Densenet-Binary Moth Flame Optimization," *International Journal of Intelligent Systems and Applications in Engineering*, vol. 12, no. 20S, pp. 647-656, 2024. [Online]. Available: https://ijisae.org/index.php/IJISAE/article/view/5195

[24] Mohammad Al-Fawa'reh et al., "MalBot-DRL: Malware Botnet Detection using Deep Reinforcement Learning in IoT Networks," *IEEE Internet of Things Journal*, vol. 11, no. 6, pp. 9610-9629, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[25] Muhammad Aamir, and Syed Mustafa Ali Zaidi, "Clustering based Semi-Supervised Machine Learning for DDoS Attack Classification," *Journal of King Saud University-Computer and Information Sciences*, vol. 33, no. 4, pp. 436-446, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[26] Tuğba Aytaç, Muhammed Ali Aydın, and Abdül Halim Zaim, "Detection DDoS Attacks using Machine Learning Methods," *Electrica*, vol. 20, no. 2, pp. 159-167, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[27] Sabah Alzahrani, and Liang Hong, "A Survey of Cloud Computing Detection Techniques Against DDoS Attacks," *Journal of Information Security*, vol. 9, no. 1, pp. 45-69, 2017. [CrossRef] [Google Scholar] [Publisher Link]

[28] Abdel-Rahman Al-Ghuwairi et al., "Intrusion Detection in Cloud Computing based on Time Series Anomalies Utilizing Machine Learning," *Journal of Cloud Computing*, vol. 12, no. 1, pp. 1-17, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[29] Abdulaziz Almazyad, Laila Halman, and Alaa Alsaeed, "Probe Attack Detection using an Improved Intrusion Detection System," *Computers, Materials and Continua*, vol. 74, no. 3, pp. 4770-4784, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[30] Waheed G. Gadallah, Nagwa M. Omar, and Hosny M. Ibrahim, "Machine Learning-based Distributed Denial of Service Attacks Detection Technique using New Features in Software-Defined Networks," *International Journal of Computer Network and Information Security*, vol. 13, no. 3, pp. 15-27, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[31] Hüseyin Polat et al., "A Novel Approach for Accurate Detection of the DdoS Attacks in SDN-based SCADA Systems based on Deep Recurrent Neural Networks," *Expert Systems with Applications*, vol. 197, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[32] N. Sivasankari, and S. Kamalakannan, "Fuzzy Logic-based Man-in-the-Middle Attack Detection and Improving Routing Efficiency in the IoT Network," *2023 International Conference on Applied Intelligence and Sustainable Computing (ICAISC)*, Dharwad, India, pp. 1-6, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[33] Metehan Gelgi et al., "Systematic Literature Review of IoT Botnet DDoS Attacks," *Sensors*, vol. 24, no. 11, pp. 1-37, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[34] Segun I. Popoola et al., "Federated Deep Learning for Zero-Day Botnet Attack Detection in IoT Edge Devices," *IEEE Internet of Things Journal*, vol. 9, no. 5, pp. 3930-3944, 2022. [CrossRef] [Google Scholar] [Publisher Link]