International Journal of Safety and Security Engineering
Vol. 15, No. 2, February, 2025, pp. 297-307
Journal homepage: http://iieta.org/journals/ijsse

Leveraging Centrality Matrix and Enhanced Walrus Evoked Learning Framework for Digital Forensics Anomaly Data Detection

Srikanth Addagatla, G. Madhukar Rao*

Department of Computer Science and Engineering, Koneru Lakshmaiah Education Foundation, Hyderabad 500075, Telangana, India

Corresponding Author Email: madhusw511@klh.edu.in

https://doi.org/10.18280/ijsse.150210

Received: 31 December 2024
Revised: 23 January 2025
Accepted: 13 February 2025
Available online: 28 February 2025

Keywords:
digital forensics (DF), artificial intelligence (AI), centrality measures, enhanced walrus evoked multi-layer neural network

ABSTRACT

Digital forensics (DF) has emerged as a crucial strategy for tracing offenders and ensuring social justice for the common man. With the increasing exploitation of digital gadgets by criminals, cybercrimes have risen significantly, affecting people's daily lives. Consequently, the primary goal of DF is to ensure that digital evidence remains unaltered by identifying, collecting, analyzing, and assessing data to reconstruct historical events. However, detecting data anomalies that indicate illegal behavior remains one of the most daunting challenges in DF. Current systems hinder DF investigations due to inaccuracies in identifying aberrant patterns in forensic data. With the advent of artificial intelligence (AI), unusual activities in DF data can now be detected more effectively. However, achieving higher accuracy remains a major challenge in this field. This research article proposes a novel hybrid learning framework that integrates centrality measures with an Enhanced Walrus Evoked Extreme Feedforward Neural Network (EW-EFNN) to identify intrusions in crime-related digital forensic data. The proposed framework is trained using different traffic data generated on computers to determine whether they have been tampered with by specific intruder programs. Real-time datasets are created using Wireshark to analyze various DF anomaly patterns, followed by centrality measures for feature extraction and an enhanced walrus-evoked learning network for optimal detection performance. Comprehensive experiments are conducted using real-time traffic data, evaluating performance metrics such as accuracy, precision, recall, specificity, and F1-score. The results demonstrate that the proposed model achieves 96% accuracy, 95.7% precision, 95.8% recall, and a 96% F1-score. Compared to existing systems, the proposed learning framework outperforms others in detecting anomaly patterns in DF data.

1. INTRODUCTION

Smart environments today provide a wide array of advanced technologies and services, including smart transportation systems, autonomous vehicles, intelligent homes, urban lighting solutions, ticketing for travel, energy-efficient grids, and sophisticated sensors [1]. The functionality of these systems is largely reliant on small-scale electronic chips and IoT devices like sensors, wireless communication technologies, Radio-Frequency Identification (RFID) devices, location-tracking systems, and Near-Field Communication (NFC) tools.

The rise of smart environments has attracted criminal activity, with offenders exploiting these digital platforms. In response, DF has become an essential method for identifying these criminals and supporting law enforcement in holding them accountable [2-4]. The process of reviewing, scrutinizing, and drawing conclusions from electronic data is regarded as DF and plays a significant role in criminal investigations [5-7].

Suspicious activities and unsafe events can cause organizations to face significant financial losses and damage their reputation. Such incidents may have severe consequences for both individuals and organizations. As indicated by statistics from published reports, there have already been over five million instances of computer-related violations [8, 9]. A substantial number of cybercrimes remain undocumented, as victims often refrain from reporting them to authorities. These victims may feel embarrassed or uncertain or perceive that the authorities are not adequately addressing the issue to hold perpetrators accountable. Moreover, the government must work to curb cybercrime by addressing the gaps in expertise and workforce competencies.

Law enforcement agencies are becoming more reliant on DF to catch and prosecute criminals, as the use of digital devices in criminal acts continues to grow [10, 11]. Identifying abnormal or suspicious behavior within digital forensics (DF) data is crucial, yet it presents a significant challenge. Designing an intelligent detection system is vital as it empowers investigators to ascertain unusual behavioral patterns that could indicate criminal activity. Traditionally, anomaly detection in DF has relied on statistical methods and rule-based systems [12]. These techniques may fail to detect subtle irregularities intended to deceive investigative frameworks.

AI, especially through Machine Learning (ML) and Deep Learning (DL) approaches, offers efficient and sophisticated methods for generating valuable insights for decision-makers. These approaches involve analysing datasets from various angles and transforming them into meaningful formats. The aim of these approaches is to learn from past data to forecast future behaviors. By leveraging ML methods, researchers can enhance their ability to identify patterns in criminal activities and gain insights from historical data regarding when, where, and how cybercrimes are likely to occur.

Several ML algorithms, such as Support Vector Neural Networks (SVNN), Support Vector Machines (SVM), Multi-layer Neural Networks, and K-Nearest Neighborhood (KNN), are deployed for the detection of anomalies in DF systems. These techniques are used in numerous fields like financial fraud detection and intrusion detection. But these algorithms need to be studied extensively so they can be applied in DF analysis. Hence, the current approaches need the brighter light of improvisation in terms of DF data. Inspired by the aforementioned drawbacks, this article aims to examine the challenges in determining anomalies in DF data pertaining to criminal activity. The paper proposes the novel ensemble of centrality-based feature extraction with an enhanced Walrus Evoked multi-layer learning framework for anomaly detection in forensic digital data. The key contributions are as follows:

(1) The paper recommends centrality feature extraction methods for better detection of anomalies in DF data.

(2) An enhanced Walrus Evoked multi-layer framework for achieving the robustness and scalability of the scheme is recommended in this research.

(3) The paper proposes a novel synthetic data generation method based on Wireshark and the Python environment to evaluate the proposed model.

(4) Extensive evaluation is facilitated by utilizing the above datasets, and various evaluation metrics are measured, analyzed, and examined with the varied schemes. From the experimental outcomes, the recommended model has outshined the other available schemes.

The structure of the study is organised in the following manner: The relevant studies by distinct authors on ML models deployed for anomaly detection in DF investigations are discussed in Section 2. The novel dataset generation, feature extraction, and learning networks are discussed in Section 3. Implementation details are provided in Section 4. Experimental outcomes and evaluative assessments are outlined in Section 5. Finally, the study concludes with suggestions for future improvements in Section 6.

2. RELATED WORKS

Wang et al. [13] proposed a novel fusion model combining Isolation Forest (IF), GAN, and transformer for Network Anomaly Detection (NAD) and log analysis. They address key challenges in handling high-dimensional data and complex network topologies through this integrated approach. The IF enables quick anomaly identification, while GAN generates synthetic training data, and transformer processes time-series data. While their experimental results demonstrate improved accuracy and reduced false alarm rates, the model's computational poses integration challenges in real-world deployments.

Nkashama et al. [14] investigated the critical issue of data contamination in DL-based NAD systems. Their research evaluates six unsupervised DL algorithms' robustness against contaminated training data. They propose an enhanced auto-encoder with constrained latent representation to improve resilience to data contamination. However, the study could benefit from broader evaluation across diverse network environments and attack scenarios to validate generalizability.

Islam et al. [15] proposed a Novel Support Vector Neural Network (NSVNN) for anomaly detection in DF. They evaluated the NSVNN against existing algorithms like SVM and neural networks using DF data. The research highlights feature importance and the decision-making process, contributing to effectiveness examining in DF. Meanwhile, the research does not discuss the scalability of the NSVNN for larger data and its effectiveness in real-time Forensics investigations.

Yang et al. [16] presented an approach that includes multi-perspective feature engineering, Unsupervised Anomaly Detection (UAD), and extensive outcome refinement processes. The framework employs a combination of Gaussian Mixture Model (GMM) While the study portrays the capability of the recommended approach on real-world evolving network data, it lacks comparison with varied cutting-edge anomaly identification techniques in the context of DF.

Jones et al. [17] proposed a scheme to recognize suspicious drug-related activity on mobile devices using Forensics and Natural Language Processing (NLP) approach. But it does not address the challenges of applying these techniques to diverse types of mobile Forensics data and discuss the model's performance on unseen data.

Ashraf et al. [18] developed a model for behaviour detection utilizing social media forensic analysis on social media accounts. Although the framework demonstrates potential benefits for cybercrime and cyber-security agencies, it does not address privacy concerns or ethical considerations in analysing social media data for behaviour detection.

Wei et al. [19] introduced a novel UAD framework utilizing Cascaded Auto Encoders (CAEs) and integrated optimization network for threat recognition. Their end-to-end framework utilizes Bidirectional Long Short-Term Memory (BiLSTM) for feature extraction and includes a hypergraph correction module to reduce false positive rates. The discussion on the framework's adaptability to varied types of organizational data is missing.

Studiawan and Sohel [20] developed a DL technique to ascertain anomalous events in a Forensics timeline using deep autoencoders. Their method preprocesses system logs, applies deep and sparse autoencoders to build a Forensics timeline, and plots the outcome using Time sketch. In experiments with four public Forensics datasets, the recommended approach achieved good results than existing approaches.

Li et al. [21] introduced Swiss Log, a reliable and integrated DL approach for identifying various anomalies in digital log data. Experiments on real-world and synthetic data highlights the efficiency of Swiss Log, the paper falls short in providing detailed performance metrics and comparisons with other anomaly detection methods.

3. PROPOSED METHODOLOGY

The proposed framework consists of four components such as synthetic data generation, data-pre-processing, centrality feature extractor and walrus optimized extreme learning networks have shown in Figure 1. The detailed description of the each and every module is as follows.

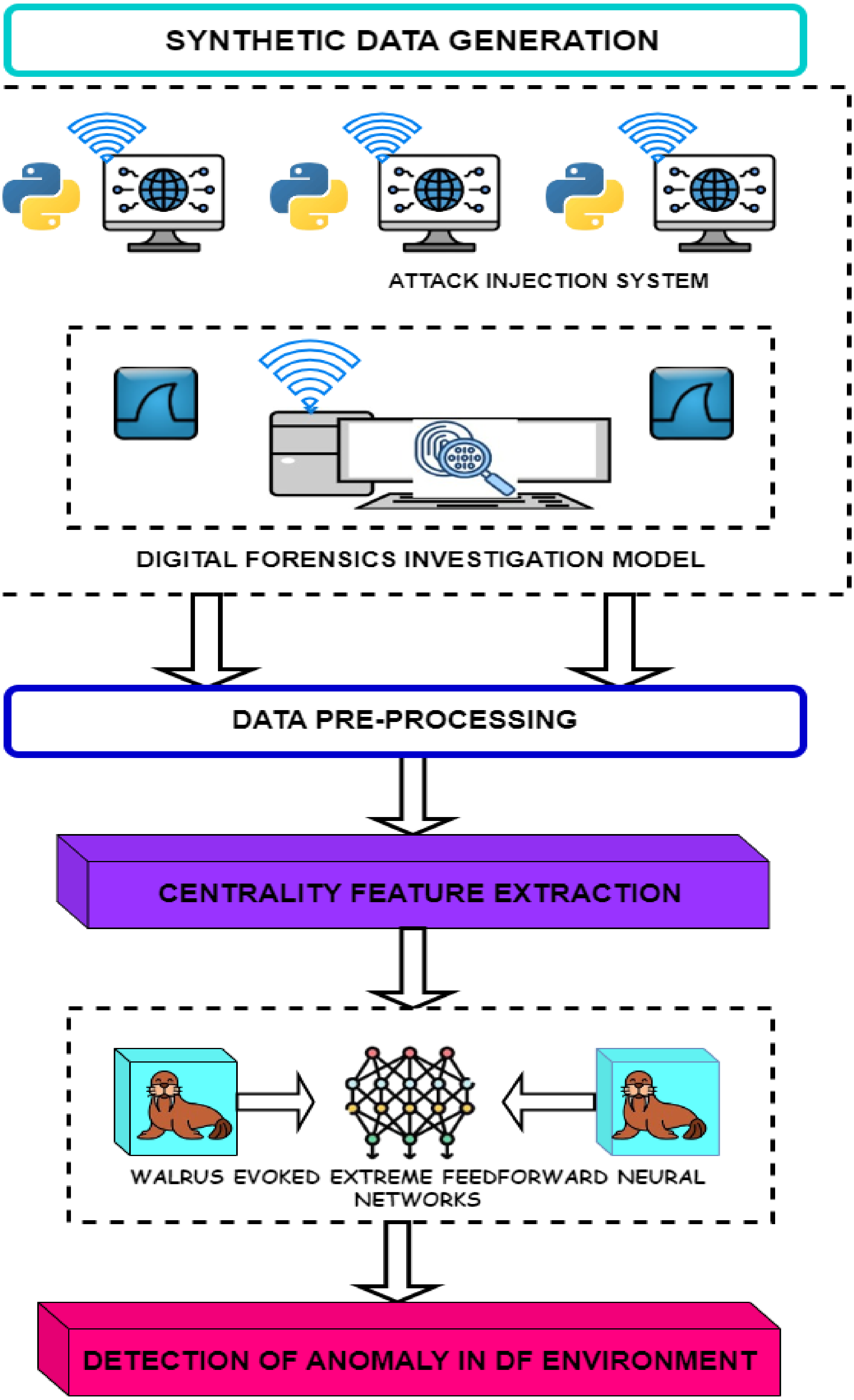

SYNTHETIC DATA GENERATION
ATTACK INJECTION SYSTEM
DIGITAL FORENSICS INVESTIGATION MODEL
DATA PRE-PROCESSING
CENTRALITY FEATURE EXTRACTION
WALRUS EVOKED EXTREME FEEDFORWARD NEURAL NETWORKS
DETECTION OF ANOMALY IN DF ENVIRONMENT

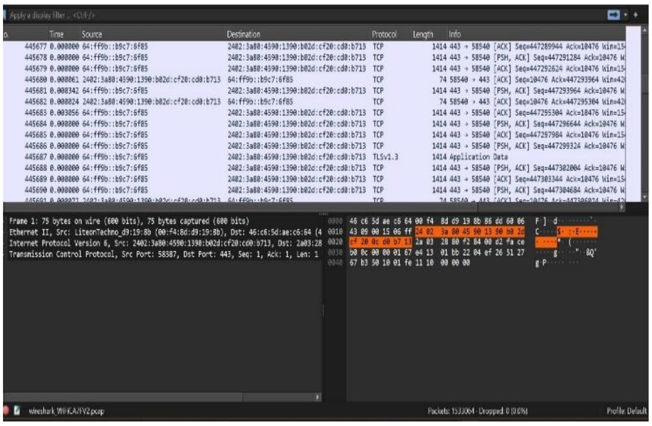
Figure 1. Overall Architectures for the recommended framework

3.1 Synthetic data generation

The research utilizes synthetic data generation to evaluate the proposed model in detecting anomalies within the DF framework. To generate the synthetic data, Wireshark, a network packet analyzer, is used to capture network packets and attempt to reveal the packet data. The detailed components of Wireshark are presented in Table 1. Figures 2-4 show the screenshots from Wireshark, capturing packets from different devices.

Table 1. Components of Wireshark used for the synthetic data generation

Sl. No.	Components of Wireshark	Description
1	T-Shark	Command Line Inputs
2	External Capture Interface	To capture the packets from the network devices
3	ETwdump	To display and capture the packets from the external devices
4	AndriodDump	To capture and display the packets from the Android Mobiles
5	SShdump, WIFIDump	To capture and display the packets from the networked devices
6	UDPdump	To display the packets from the UDP mechanism



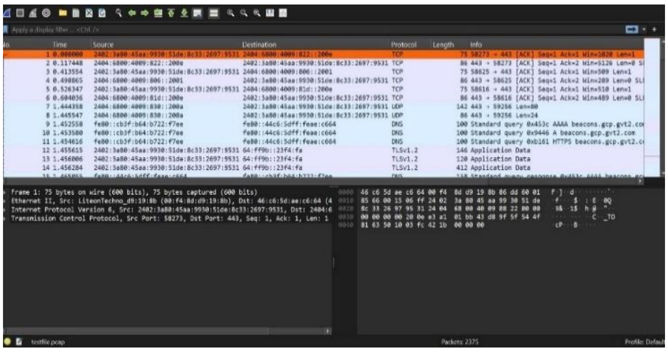
```
Time     Source                                     Destination                              Protocol Length Info
445677 0.000000 64:ff9b::b9c7:6f85                   2402:3a80:4590:1390:b02d:cf20:cd0:b713   TCP     1414 443 → 58540 [ACK] Seq=447289944 Ack=10476 Win=15
445678 0.000000 64:ff9b::b9c7:6f85                   2402:3a80:4590:1390:b02d:cf20:cd0:b713   TCP     1414 443 → 58540 [PSH, ACK] Seq=447291284 Ack=10476 W
445679 0.000000 64:ff9b::b9c7:6f85                   2402:3a80:4590:1390:b02d:cf20:cd0:b713   TCP     1414 443 → 58540 [ACK] Seq=447292624 Ack=10476 Win=15
445680 0.000061 2402:3a80:4590:1390:b02d:cf20:cd0:b713 64:ff9b::b9c7:6f85                      TCP       74 58540 → 443 [ACK] Seq=10476 Ack=447293964 Win=42
445681 0.000342 64:ff9b::b9c7:6f85                   2402:3a80:4590:1390:b02d:cf20:cd0:b713   TCP     1414 443 → 58540 [PSH, ACK] Seq=447293964 Ack=10476 W
445682 0.000024 2402:3a80:4590:1390:b02d:cf20:cd0:b713 64:ff9b::b9c7:6f85                      TCP       74 58540 → 443 [ACK] Seq=10476 Ack=447295304 Win=42
445683 0.003056 64:ff9b::b9c7:6f85                   2402:3a80:4590:1390:b02d:cf20:cd0:b713   TCP     1414 443 → 58540 [ACK] Seq=447295304 Ack=10476 Win=15
445684 0.000000 64:ff9b::b9c7:6f85                   2402:3a80:4590:1390:b02d:cf20:cd0:b713   TCP     1414 443 → 58540 [PSH, ACK] Seq=447296644 Ack=10476 W
445685 0.000000 64:ff9b::b9c7:6f85                   2402:3a80:4590:1390:b02d:cf20:cd0:b713   TCP     1414 443 → 58540 [ACK] Seq=447297984 Ack=10476 Win=15
445686 0.000000 64:ff9b::b9c7:6f85                   2402:3a80:4590:1390:b02d:cf20:cd0:b713   TCP     1414 443 → 58540 [PSH, ACK] Seq=447299324 Ack=10476 W
445687 0.000000 64:ff9b::b9c7:6f85                   2402:3a80:4590:1390:b02d:cf20:cd0:b713   TLSv1.3 1414 Application Data
445688 0.000000 64:ff9b::b9c7:6f85                   2402:3a80:4590:1390:b02d:cf20:cd0:b713   TCP     1414 443 → 58540 [PSH, ACK] Seq=447302004 Ack=10476 W
445689 0.000000 64:ff9b::b9c7:6f85                   2402:3a80:4590:1390:b02d:cf20:cd0:b713   TCP     1414 443 → 58540 [ACK] Seq=447303344 Ack=10476 Win=15
445690 0.000000 64:ff9b::b9c7:6f85                   2402:3a80:4590:1390:b02d:cf20:cd0:b713   TCP     1414 443 → 58540 [PSH, ACK] Seq=447304684 Ack=10476 W

Frame 1: 75 bytes on wire (600 bits), 75 bytes captured (600 bits)
Ethernet II, Src: LiteonTechno_d9:19:8b (00:f4:8d:d9:19:8b), Dst: 46:c6:5d:ae:c6:64 (4
Internet Protocol Version 6, Src: 2402:3a80:4590:1390:b02d:cf20:cd0:b713, Dst: 2a03:28
Transmission Control Protocol, Src Port: 58387, Dst Port: 443, Seq: 1, Ack: 1, Len: 1

0000  46 c6 5d ae c6 64 00 f4  8d d9 19 8b 86 dd 60 06
0010  43 09 00 15 06 ff 24 02  3a 80 45 90 13 90 b0 2d
0020  cf 20 0c d0 b7 13 2a 03  28 80 f2 84 00 d2 fa ce
0030  b0 8c 00 00 01 67 e4 13  01 bb 22 04 ef 26 51 27
0040  67 b3 50 10 01 fe 11 10  00 00 00
```

Apply a display filter ... <Ctrl-/>
Packets: 153064 · Dropped: 0 (0.0%)
Profile: Default

Figure 2. Wireshark-capturing the packets from the networked devices



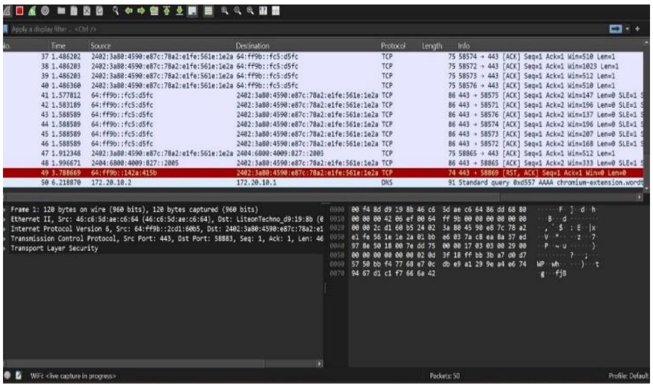
```
No. Time     Source                                    Destination                               Protocol Length Info
 1 0.000000  2402:3a80:45aa:9930:51de:8c33:2697:9531   2404:6800:4009:822::200e                  TCP       75 58273 → 443 [ACK] Seq=1 Ack=1 Win=1020 Len=1
 2 0.117448  2404:6800:4009:822::200e                  2402:3a80:45aa:9930:51de:8c33:2697:9531   TCP       86 443 → 58273 [ACK] Seq=1 Ack=2 Win=5126 Len=0 S
 3 0.413554  2402:3a80:45aa:9930:51de:8c33:2697:9531   2404:6800:4009:806::2001                  TCP       75 58625 → 443 [ACK] Seq=1 Ack=1 Win=509 Len=1
 4 0.498865  2404:6800:4009:806::2001                  2402:3a80:45aa:9930:51de:8c33:2697:9531   TCP       86 443 → 58625 [ACK] Seq=1 Ack=2 Win=289 Len=0 SL
 5 0.526347  2402:3a80:45aa:9930:51de:8c33:2697:9531   2404:6800:4009:81d::200e                  TCP       75 58616 → 443 [ACK] Seq=1 Ack=1 Win=510 Len=1
 6 0.604036  2404:6800:4009:81d::200e                  2402:3a80:45aa:9930:51de:8c33:2697:9531   TCP       86 443 → 58616 [ACK] Seq=1 Ack=2 Win=489 Len=0 SL
 7 1.444358  2404:6800:4009:830::200a                  2402:3a80:45aa:9930:51de:8c33:2697:9531   UDP      142 443 → 59256 Len=80
 8 1.445547  2404:6800:4009:830::200a                  2402:3a80:45aa:9930:51de:8c33:2697:9531   UDP       86 443 → 59256 Len=24
 9 1.452558  fe80::cb3f:b64:b722:f7ee                  fe80::44c6:5dff:feae:c664                 DNS      100 Standard query 0x453c AAAA beacons.gcp.gvt2.co
10 1.453580  fe80::cb3f:b64:b722:f7ee                  fe80::44c6:5dff:feae:c664                 DNS      100 Standard query 0x9446 A beacons.gcp.gvt2.com
11 1.454616  fe80::cb3f:b64:b722:f7ee                  fe80::44c6:5dff:feae:c664                 DNS      100 Standard query 0xb161 HTTPS beacons.gcp.gvt2.c
12 1.455615  2402:3a80:45aa:9930:51de:8c33:2697:9531   64:ff9b::23f4:fa                          TLSv1.2  146 Application Data
13 1.456806  2402:3a80:45aa:9930:51de:8c33:2697:9531   64:ff9b::23f4:fa                          TLSv1.2  120 Application Data
14 1.456284  2402:3a80:45aa:9930:51de:8c33:2697:9531   64:ff9b::23f4:fa                          TLSv1.2  412 Application Data

Frame 1: 75 bytes on wire (600 bits), 75 bytes captured (600 bits)
Ethernet II, Src: LiteonTechno_d9:19:8b (00:f4:8d:d9:19:8b), Dst: 46:c6:5d:ae:c6:64 (4
Internet Protocol Version 6, Src: 2402:3a80:45aa:9930:51de:8c33:2697:9531, Dst: 2404:6
Transmission Control Protocol, Src Port: 58273, Dst Port: 443, Seq: 1, Ack: 1, Len: 1

0000  46 c6 5d ae c6 64 00 f4  8d d9 19 8b 86 dd 60 01
0010  85 66 00 15 06 ff 24 02  3a 80 45 aa 99 30 51 de
0020  8c 33 26 97 95 31 24 04  68 00 40 09 08 22 00 00
0030  00 00 00 00 20 0e e3 a1  01 bb 43 d8 9f 5f 54 4f
0040  81 63 50 10 03 fc 42 1b  00 00 00
```

Figure 3. Filtering the data using the Wireshark installed devices

To inject the anomalies, Python programming was employed. Multiple attack models were designed and injected from neighboring devices using Wi-Fi and Ethernet networks. Several research works [22-24] have demonstrated that DF data are vulnerable to various types of attacks. The categorization of significant anomalies is presented as follows.



```
No. Time     Source                                     Destination                                Protocol Length Info
37 1.486202  2402:3a80:4590:e87c:78a2:e1fe:561e:1e2a    64:ff9b::fc5:d5fc                          TCP       75 58574 → 443 [ACK] Seq=1 Ack=1 Win=510 Len=1
38 1.486203  2402:3a80:4590:e87c:78a2:e1fe:561e:1e2a    64:ff9b::fc5:d5fc                          TCP       75 58572 → 443 [ACK] Seq=1 Ack=1 Win=1023 Len=1
39 1.486203  2402:3a80:4590:e87c:78a2:e1fe:561e:1e2a    64:ff9b::fc5:d5fc                          TCP       75 58573 → 443 [ACK] Seq=1 Ack=1 Win=512 Len=1
40 1.486360  2402:3a80:4590:e87c:78a2:e1fe:561e:1e2a    64:ff9b::fc5:d5fc                          TCP       75 58576 → 443 [ACK] Seq=1 Ack=1 Win=510 Len=1
41 1.577812  64:ff9b::fc5:d5fc                          2402:3a80:4590:e87c:78a2:e1fe:561e:1e2a    TCP       86 443 → 58575 [ACK] Seq=1 Ack=2 Win=147 Len=0 SLE=1 S
42 1.583189  64:ff9b::fc5:d5fc                          2402:3a80:4590:e87c:78a2:e1fe:561e:1e2a    TCP       86 443 → 58571 [ACK] Seq=1 Ack=2 Win=196 Len=0 SLE=1 S
43 1.588589  64:ff9b::fc5:d5fc                          2402:3a80:4590:e87c:78a2:e1fe:561e:1e2a    TCP       86 443 → 58576 [ACK] Seq=1 Ack=2 Win=137 Len=0 SLE=1 S
44 1.588589  64:ff9b::fc5:d5fc                          2402:3a80:4590:e87c:78a2:e1fe:561e:1e2a    TCP       86 443 → 58574 [ACK] Seq=1 Ack=2 Win=196 Len=0 SLE=1 S
45 1.588589  64:ff9b::fc5:d5fc                          2402:3a80:4590:e87c:78a2:e1fe:561e:1e2a    TCP       86 443 → 58573 [ACK] Seq=1 Ack=2 Win=207 Len=0 SLE=1 S
46 1.588589  64:ff9b::fc5:d5fc                          2402:3a80:4590:e87c:78a2:e1fe:561e:1e2a    TCP       86 443 → 58572 [ACK] Seq=1 Ack=2 Win=168 Len=0 SLE=1 S
47 1.912348  2402:3a80:4590:e87c:78a2:e1fe:561e:1e2a    2404:6800:4009:827::2005                   TCP       75 58865 → 443 [ACK] Seq=1 Ack=1 Win=512 Len=1
48 1.996671  2404:6800:4009:827::2005                   2402:3a80:4590:e87c:78a2:e1fe:561e:1e2a    TCP       86 443 → 58865 [ACK] Seq=1 Ack=2 Win=333 Len=0 SLE=1 S
49 3.788669  64:ff9b::142a:415b                         2402:3a80:4590:e87c:78a2:e1fe:561e:1e2a    TCP       74 443 → 58869 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
50 6.218870  172.20.10.2                                172.20.10.1                                DNS       91 Standard query 0xd557 AAAA chromium-extension.wordt

Frame 1: 120 bytes on wire (960 bits), 120 bytes captured (960 bits)
Ethernet II, Src: 46:c6:5d:ae:c6:64 (46:c6:5d:ae:c6:64), Dst: LiteonTechno_d9:19:8b (0
Internet Protocol Version 6, Src: 64:ff9b::2cd1:60b5, Dst: 2402:3a80:4590:e87c:78a2:e1
Transmission Control Protocol, Src Port: 443, Dst Port: 58883, Seq: 1, Ack: 1, Len: 46
Transport Layer Security

0000  00 f4 8d d9 19 8b 46 c6  5d ae c6 64 86 dd 68 80
0010  00 00 00 42 06 ef 00 64  ff 9b 00 00 00 00 00 00
0020  00 00 2c d1 60 b5 24 02  3a 80 45 90 e8 7c 78 a2
0030  e1 fe 56 1e 1e 2a 01 bb  e6 03 7a c8 ea 8a 37 ed
0040  97 8e 50 18 00 7e dd 75  00 00 17 03 03 00 29 00
0050  00 00 00 00 00 00 02 0d  3f 18 ff bb 3b a7 d0 d7
0060  57 50 bb f4 77 68 e7 0c  db e9 a1 29 9e a4 e6 74
0070  94 67 d1 c1 f7 66 6a 42
```

Apply a display filter ... <Ctrl-/>
WiFi: <live capture in progress>
Packets: 50
Profile: Default

Figure 4. Anomaly data injected from devices and transmitted to Wireshark

3.1.1 Denial of Service (DoS)

A DoS attack occurs when excessive unwanted traffic originates from a single source or receiver. In this scenario, the attacker overwhelms the target with numerous malicious packets, rendering its services inaccessible to other users [25].

3.1.2 Distributed Denial of Service/Botnet attacks (DDoS)

These attacks focus on compromising a large number of vulnerable IoT devices, enabling the attacker to execute a much more potent DoS attack or other forms of assault [26].

3.1.3 Man-in-the-Middle (MitM)

This attack compromises the communication channel between the IoT device and its target data recipient. While the connection is breached, the attacker acts as an intermediary, allowing them to observe, insert, and alter the data being transmitted [27].

3.1.4 Data type Probing (D.P.)

The Data Probing attack involves altering the raw data packets as they travel between departure and arrival points [28].

Algorithm-1 presents the pseudocode for the data collection mechanism.

```
Sl. No.  Algorithm-1//Data Collection Mechanism
1        Input: Packets from the Different Sources of Devices
2        Outputs: Anomaly Data and Normal packets
3        While (True)
4            For t=1: 100000secs; t=time frame
5                Capture the Normal packets from the devices
6                Store it as the PCAP CSV files
7            End for
8            For t=100000:2000000secs; t=time frame
9                Capture the Anomaly data from the devices
10               Store it as the PCAP CSV files
11           End for
12           Go to Step 6
13           Go to Step 8
14       End While
```

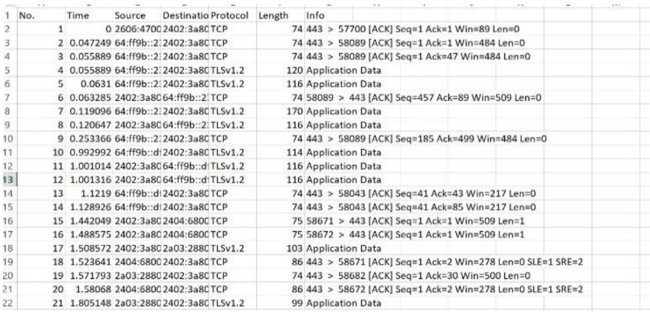
3.1.5 Spying (SP)

In this attack, an intruder exploits system vulnerabilities and uses a backdoor channel to infiltrate the system, exposing sensitive information [29].

For efficient data collection, a series of malicious anomalies were documented over a span of three weeks for further predictive analysis. To ensure a thorough evaluation of the proposed framework, it is essential to create an expanded dataset that includes both benign and harmful data. The Python API developed was used to implement various types of attacks, and the information samples collected during the experimentation are presented in Table 2. All the data were captured and stored in PCAP CSV files in Wireshark, as shown in Figure 5.

Table 2. Dataset collected during experimentation

Data-Type Descriptions	Count of Samples Gathered
Normal Data	14,90738
Malicious Data-DoS Attack	890392
Malicious Data-DDoS Attack	378904
Data type Probing Data	250895
Spying Data	109034
MIM Data	105678



1	No.	Time	Source	Destinatio	Protocol	Length	Info
2	1	0	2606:4700	2402:3a80	TCP	74	443 > 57700 [ACK] Seq=1 Ack=1 Win=89 Len=0
3	2	0.047249	64:ff9b::2	2402:3a80	TCP	74	443 > 58089 [ACK] Seq=1 Ack=1 Win=484 Len=0
4	3	0.055889	64:ff9b::2	2402:3a80	TCP	74	443 > 58089 [ACK] Seq=1 Ack=47 Win=484 Len=0
5	4	0.055889	64:ff9b::2	2402:3a80	TLSv1.2	120	Application Data
6	5	0.0631	64:ff9b::2	2402:3a80	TLSv1.2	116	Application Data
7	6	0.063285	2402:3a80	64:ff9b::2	TCP	74	58089 > 443 [ACK] Seq=457 Ack=89 Win=509 Len=0
8	7	0.119096	64:ff9b::2	2402:3a80	TLSv1.2	170	Application Data
9	8	0.120647	2402:3a80	64:ff9b::2	TLSv1.2	116	Application Data
10	9	0.253366	64:ff9b::2	2402:3a80	TCP	74	443 > 58089 [ACK] Seq=185 Ack=499 Win=484 Len=0
11	10	0.992992	64:ff9b::d	2402:3a80	TLSv1.2	114	Application Data
12	11	1.001014	2402:3a80	64:ff9b::d	TLSv1.2	116	Application Data
13	12	1.001316	2402:3a80	64:ff9b::d	TLSv1.2	116	Application Data
14	13	1.1219	64:ff9b::d	2402:3a80	TCP	74	443 > 58043 [ACK] Seq=41 Ack=43 Win=217 Len=0
15	14	1.128926	64:ff9b::d	2402:3a80	TCP	74	443 > 58043 [ACK] Seq=41 Ack=85 Win=217 Len=0
16	15	1.442049	2402:3a80	2404:6800	TCP	75	58671 > 443 [ACK] Seq=1 Ack=1 Win=509 Len=1
17	16	1.488575	2402:3a80	2404:6800	TCP	75	58672 > 443 [ACK] Seq=1 Ack=1 Win=509 Len=1
18	17	1.508572	2402:3a80	2a03:2880	TLSv1.2	103	Application Data
19	18	1.523641	2404:6800	2402:3a80	TCP	86	443 > 58671 [ACK] Seq=1 Ack=2 Win=278 Len=0 SLE=1 SRE=2
20	19	1.571793	2a03:2880	2402:3a80	TCP	74	443 > 58682 [ACK] Seq=1 Ack=30 Win=500 Len=0
21	20	1.58068	2404:6800	2402:3a80	TCP	86	443 > 58672 [ACK] Seq=1 Ack=2 Win=278 Len=0 SLE=1 SRE=2
22	21	1.805148	2a03:2880	2402:3a80	TLSv1.2	99	Application Data

Figure 5. PCAP CSV file format stored in Wireshark after the pre-defined time stamps

3.2 Data pre-processing technique

It is considered the paramount technique in designing an efficient learning model and includes data cleaning, conversion, and scaling. As the first step, missing values and null values (NaN) are eliminated from the datasets. In the second step, categorical features are converted into numerical values using the Label Encoding technique, which is included as a separate library in Scikit-Learn. Finally, data normalization is applied to the datasets, converting all the inputs to a standard scale, which is essential to avoid misinterpretations during model training. In this research, the Min-Max scaling process is used, which converts each feature to a predefined range from 0 to 1.

3.3 Centrality feature extraction

The primary goal of this research is to design efficient attribute databases capable of distinguishing between normal and influential nodes. Various methods for detecting centrality measures have been proposed in previous literature to signify the significance of nodes. However, this paper emphasizes the application of an expanded set of centrality metrics to achieve the most precise classification of influential nodes.

3.3.1 Degree centralities

This refers to the number of connections linked to the nodes within a network. It comprises two variations: indegree centrality and outdegree centrality. The values of these centralities can be calculated using the equations provided below.

Indegree centrality

$$D_{in}(P_i) = \left|P_{ji} \in P\right|, j \neq i \tag{1}$$

P_{ji} is the edge going from P_i node to examined node P.

Outdegree centrality

$$D_{ot}(P_i) = \left|P_{ij} \in P\right|, i \neq j \tag{2}$$

3.3.2 Amongness centralities

Amongness centrality indicates the proportion of all shortest paths that pass through specific nodes. The formula for calculating betweenness centrality is presented below:

$$D_B(P_i) = \sum_{P_S \neq P_i \neq P_d} \frac{\mu_{P_s,P_d}(P_i)}{\mu_{P_s,P_d}} \tag{3}$$

where, $\mu_{P_s,P_d}(P_i)$ is the shortest paths among nodes P_S and P_d relaying through node Pi and μ_{P_s,P_d}.

3.3.3 Closeness centralities

This signifies the distance of nodes in the network, expressed mathematically as follows:

$$D_c(P_i) = \frac{N}{\sum_{Py} d(P_y, P_i)} \tag{4}$$

3.3.4 Eigen vector centralities

The eigenvector centrality method is used to derive the centralities of other nodes in the network. The formula for determining this is given below:

$$E_v(P_i) = {}^{1}/\ A \sum_{k} \gamma_{P_k,P_i} * E_v(P_k) \tag{5}$$

where, $A = \alpha(k, i)$ is the adjacency matrix of a graph and γ a constant.

3.3.5 PageRank centralities

PageRank centrality ranks nodes based on their centrality within networks, with the expression used to calculate it given by:

$$R_p(P_i) = \rho \sum_{k} \frac{A_{P_k,P_i}}{d_k} * R_p(P_k) + \beta \tag{6}$$

where, ρ and β are constant values, and d_k indicates the out-degree of node P_k if the degree is greater than zero; if the out-degree of node P_k is zero, then d_k is set to 1. Furthermore, A=(ai,j) signifies the adjacency matrix of a graph, where A=α(k,i) serves as the adjacency matrix.

3.3.6 Position centrality

This is considered the most important measure, representing the position of nodes relative to key nodes identified by the PageRank method.

$$\mathrm{S_c(U_i)} = \mathrm{B} \sum_{\mathrm{k}} \gamma_{\mathrm{U_i,U_k}} * R_U(U_i) \tag{7}$$

where, B=(ai,j) is the adjacency matrix of a graph and $R_U(U_i)$ is the Page Rank of the node.

3.3.7 Clustering co-efficient

This factor denotes the fraction of triangles found among the residing triangles in the nodes' neighborhood. The quantitative expression for calculating the clustering coefficient is given by:

$$C_c=2M_{U,i}/O_i(O_i-1) \tag{8}$$

Apart from these centralities, this paper also discusses another important parameter called Clique, which plays a crucial role in improving the classifier's performance. Reference [30] contains additional information regarding this measure, which is incorporated into the recommended scheme. The methods for assessing K-Score and K-Shell are explored in reference [31].

Another crucial metric used for classification includes neighborhood variability and timestamp centrality.

All attribute vectors utilized for classification are detailed in Table 3.

Table 3. Summary of centrality feature extraction mechanism

Centrality Features	Significance
In degree Centrality Out degree Centrality	Signifies the number of connections tied to the nodes.
Amongness Centrality	Indicates the proportion of all shortest paths that traverse the nodes.
Closeness Centrality	Illustrates the spatial relationships of nodes in the networks.
Eigen Vector Centrality	Utilized for calculating the centrality measures of other nodes in the network
PageRank Centrality	Evaluates the hierarchy of nodes according to their centrality within the networks
Position Centrality	Indicates the location of nodes in relation to significant nodes.
Clustering Co-efficient	Indicates the proportion of triangles that exist relative to all triangles in the neighborhood of the nodes.
K-Shell Centrality	Indicates the K count for decomposing networks.
K-Score Centrality	Determines the K number of eliminated nodes in the networks.
Time Stamp Centrality	Assessment of the time lag among transmitted and received messages in the network.

3.4 Walrus evoked multilayer network classification

This section discusses the working principles of the Walrus Optimization Algorithm (WaOA) and the proposed model.

3.4.1 Walrus Optimization Algorithm (WaOA)

The walrus is a substantial aquatic species characterized by flippers and a patchy dispersion in the Arctic Ocean and subarctic regions surrounding the North Pole. The detailed characteristics of living habitat and its hunting nature are detailed in previous study [32]. Conversely, walruses can defend themselves with their tusks during such encounters.

(i) Directing individuals to follow the lead of a walrus possessing the longest tusks. Identifying the top member of the population during the search phase guides the algorithm toward favorable regions. In walrus social structures, the most dominant individual, recognized by its extended tusks, plays an essential role in leading the group. The movement of these walruses causes notable shifts in their positions. Mimicking these substantial relocations enhances the method's global search capabilities and exploration efficiency.

(ii) The walrus's migration towards shores is a natural behaviour triggered by the warming climate during summer.

During this process, walruses significantly alter their locations, often moving toward outcrops or rocky coastlines. In the WaOA simulation, the positions of various walruses are treated as potential migration targets. One of these locations is chosen at random, and the walrus navigates toward it. By emulating this strategy, WaOA improves its global search and exploration abilities. Unlike the foraging behavior driven by the dominant walrus, the migration strategy allows for a population update mechanism that does not rely on any specific individual, such as the top-contributing member. This updating method helps avoid premature convergence and prevents the algorithm from becoming trapped in local optima.

(iii) Defending or evading predators. Walruses defend themselves against natural predators, including polar bears and killer whales, using a prolonged pursuit tactic. This pursuit occurs within a limited zone surrounding the walrus, resulting in minor positional shifts. By simulating these subtle movements during confrontations, the WaOA enhances its capacity for local searching and exploitation, enabling it to converge on superior solutions.

3.4.2 Algorithm initialization

This algorithm is primarily based on the walrus population. It represents a potential remedy to the optimization challenge at hand. Consequently, the location of each walrus within the search space represents the potential values for the variables in the problem. During the initial stages of implementing WaOA, the walrus populations are generated randomly. This population matrix for WaOA is established using Eq. (9):

$$X = \begin{bmatrix} X_1 \\ \vdots \\ X_i \\ \vdots \\ X_N \end{bmatrix}_{N\times m} = \begin{bmatrix} f_{1,1} & \cdots & f_{1,j} & \cdots & f_{1,m} \\ \vdots & \ddots & \vdots & \dots & \vdots \\ f_{i,1} & \cdots & f_{i,j} & \cdots & f_{i,m} \\ \vdots & \ddots & \vdots & \dots & \vdots \\ f_{N,1} & \cdots & f_{N,j} & \cdots & f_{N,m} \end{bmatrix}_{N\times m}, \tag{9}$$

X represents the population of walruses, where X_i denotes the i-th walrus (representing a potential solution), and $f_{i,j}$ signifies the j-th decision proposed by the i-th walrus. Here, N indicates the total number of walruses, while mmm refers to the total count of decision variables. The calculated score for the objective function (OF) derived from the walruses are outlined in Eq. (10).

$$F = \begin{bmatrix} F_1 \\ \vdots \\ F_i \\ \vdots \\ F_N \end{bmatrix}_{N\times 1} = \begin{bmatrix} F(X_1) \\ \vdots \\ F(X_i) \\ \vdots \\ F(X_N) \end{bmatrix}_{N\times 1}, \tag{10}$$

F represents the vector of OFs, while F_i denotes the evaluated score of the OF based on the ith walrus. The values of the OF serve as the most effective indicators for assessing the quality of potential solutions. The potential solution yielding the highest value for the OF is referred to as the best member, whereas the solution that produces the lowest value is termed the worst member.

3.4.3 Phase 1: Exploration phase

Walruses exhibit a varied diet, consuming over sixty types of marine species. Nonetheless, they particularly favor benthic bivalves, especially clams, which they locate by foraging along the ocean floor. They search for food by utilizing vigorous flipper movements and their sensitive vibrissae. In this foraging process, the most robust walrus with the longest tusks leads other members of the group to food sources. The tusk length in walruses correlates with the integrity of the OF values of potential solutions. Thus, the optimal candidate solution, characterized by the best OF value, is regarded as the strongest walrus among the group. This foraging behavior allows walruses to cover several areas of the search space, enhancing the exploration abilities of the WaOA approach during global searches. The method for updating walrus locations is quantitatively designed based on their feeding habits, supervised by the leading member of the group, as represented in Eqs. (11) and (12):

$$f_{i,j}^{P_1} = f_{i,j} + rand_{i,j} \cdot (SW_j - I_{i,j} \cdot f_{i,j}) \quad (11)$$

$$X_i = \{ \begin{matrix} X_i^{P_1}, & F_i^{P_1} < F_i, \\ X_i, & else, \end{matrix} \quad (12)$$

where, $X_i^{P_1}$ represents the recently instituted location for the ith walrus during the first phase, $f_{i,j}^{P_1}$ indicates its jth dimension, and $F_i^{P_1}$ denotes its corresponding OF value. The terms $rand_{i,j}$ are random values drawn within the limit [0, 1], while SW signifies the optimal candidate solution regarded as the most capable walrus. The integers $I_{i,j}$ are randomly selected to be either 1 or 2, which enhances the exploration capacity of the algorithm. Specifically, if $I_{i,j}$ equals 2, it results in more substantial and extensive alterations to the walrus positions, examined to a value of 1, which represents the typical state of displacement. These parameters aid in enhancing the global search efficacy of the algorithm, enabling it to escape local optima and effectively locate the true optimal region within the problem-solving landscape.

3.4.4 Phase 2: Migration

One inherent behavior of walruses is their movement towards rocky outcrops or beaches as summer progresses and temperatures rise. This migratory behavior is utilized in the WaOA to help walruses explore the search space for optimal locations. This behavioral pattern is mathematically represented using Eqs. (13) and (14). According to Eq. (14), if this new position enhances the OF value, it replaces the walrus's prior location.

$$f_{i,j}^{P_2} = \{ \begin{matrix} f_{i,j} + rand_{i,j} \cdot (f_{k,j} - I_{i,j} \cdot f_{i,j}), F_k < F_i; \\ f_{i,j} + rand_{i,j} \cdot (f_{i,j} - f_{k,j}), else, \end{matrix} \quad (13)$$

$$X_i = \{ \begin{matrix} X_i^{P_2}, & F_i^{P_2} < F_i; \\ X_i, & else, \end{matrix} \quad (14)$$

where, $X_i^{P_2}$ denotes the upgraded position of the ith walrus based on phase two, $f_{i,j}^{P_2}$ represents its jth dimension, and $F_i^{P_2}$ denotes its OF value. $X_k, k \in \{1,2,\dots,N\}$ and $k \neq i$ signifies the locations of the choosen walruses that the ith walrus migrates towards, while $f_{k,j}$is its jth dimension and F_k is its corresponding OF value.

3.4.5 Phase 3: Exploitation phase

Walruses often encounter dangers from polar bears and orcas. Their tactics for escaping and confronting these predators lead to changes in their locations within their habitat. By imitating these natural activities, the WaOA enhances its capability to exploit local search areas around potential solutions. This positional change among walruses is modeled as occurring within a walrus-centred neighborhood defined by a specific radius. Initially, the algorithm prioritizes a global search to identify optimal regions within the search space, which is why the radius is considered to be variable-beginning at a maximum value and gradually decreasing with each iteration of the algorithm. Consequently, local lower and upper bounds are utilized in the segment of the WaOA to adjust the radius dynamically through successive algorithm runs. To mimic its behavior, a neighborhood is established around them, wherein a new location is randomly constructed within this vicinity.

$$f_{i,j}^{P_3} = f_{i,j} + \left(lb_{local,j}^{t} + \left(ub_{local,j}^{t} - rand \cdot lb_{local,j}^{t}\right)\right) (15) \quad (15)$$

$$Localbounds: \{ \begin{matrix} lb_{local,j}^{t} = \frac{lb_j}{t}, \\ ub_{local,j}^{t} = \frac{ub_j}{t}, \end{matrix} \quad (16)$$

$$X_i = \{ \begin{matrix} X_i^{P_3}, & F_i^{P_3} < F_i; \\ X_i, & else, \end{matrix} \quad (17)$$

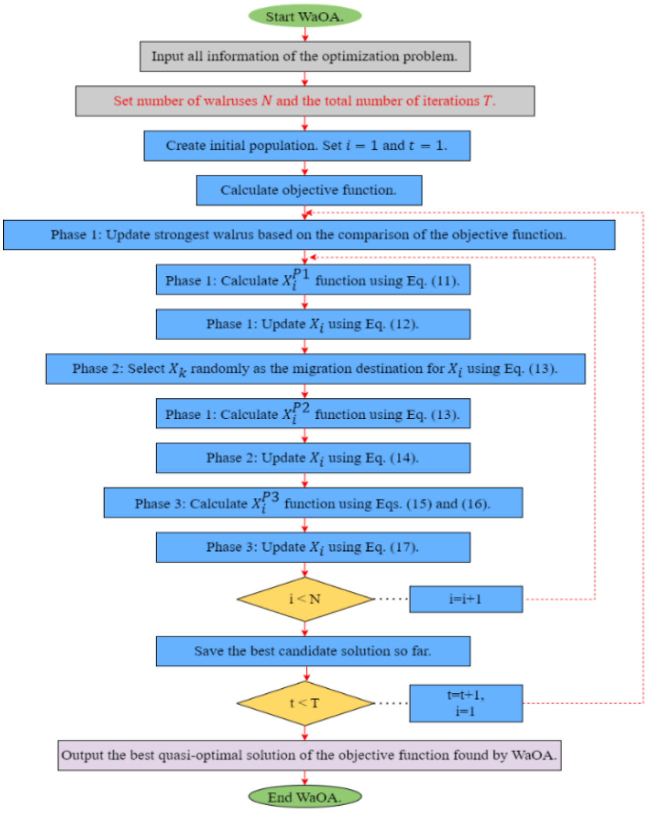

Start WaOA.
Input all information of the optimization problem.
Set number of walruses N and the total number of iterations T.
Create initial population. Set i = 1 and t = 1.
Calculate objective function.
Phase 1: Update strongest walrus based on the comparison of the objective function.
Phase 1: Calculate X_i^{P1} function using Eq. (11).
Phase 1: Update X_i using Eq. (12).
Phase 2: Select X_k randomly as the migration destination for X_i using Eq. (13).
Phase 1: Calculate X_i^{P2} function using Eq. (13).
Phase 2: Update X_i using Eq. (14).
Phase 3: Calculate X_i^{P3} function using Eqs. (15) and (16).
Phase 3: Update X_i using Eq. (17).
i < N
i=i+1
Save the best candidate solution so far.
t < T
t=t+1, i=1
Output the best quasi-optimal solution of the objective function found by WaOA.
End WaOA.

Figure 6. Flowchart of the WaOA model used for the tuning the network

3.4.6 Repetition process

After modifying the locations of the walruses following the first, second, and third phases, the preliminary iteration of the WaOA is concluded. New scores for the locations of the walruses and the OF are computed. Once the algorithm execution is complete, WaOA presents the most optimal candidate solution identified throughout the procedure as the resolution to the specified problem. The WaOA flow diagram is outlined in Figure 6.

3.5 Extreme feed forward training networks

The proposed model incorporates the concept of Extreme Learning Machines (ELM) as introduced by Huang et al. [33], aiming for rapid and precise classification of various grades. It employs a single hidden layer that does not require mandatory tuning. ELM leverages kernel functions to attain high accuracy and improved efficiency. One of the primary benefits of ELM is its ability to minimize training errors while providing superior approximation. Due to its automatic adjustment of weight biases and utilization of non-zero activation functions, ELM is widely used for classification tasks and deriving categorization values.

In this framework, the 'E' neurons within the hidden layer must utilize an activation function that is highly differentiable, such as the sigmoid function, while the output layer employs a linear function. Notably, the hidden layers in ELM do not require mandatory tuning. The weights for the hidden layer can be assigned arbitrarily, including the bias weights.

$$f_L(x) = \sum_{i=1}^{L} \beta_i l_i(v) = l(v)\beta \tag{18}$$

where, x: input features from encoder-decoder.

β: output weight vector

$$\beta = [\beta_1, \beta_2, \ldots\ldots\ldots\ldots.\beta_L]^T \tag{19}$$

l(v): output hidden layer as shown in the equation below:

$$l(v) = [l_1(v), l_2(v), \ldots\ldots\ldots\ldots..l_L(v) \tag{20}$$

To ascertain Output Vector O, denotes target vector, the hidden layer E is expressed by Eq. (21).

$$E = \begin{bmatrix} l(v_1) \\ l(v_2) \\ \vdots \\ l(v_N) \end{bmatrix} \tag{21}$$

The foundational execution of the ELM makes use of the minimal non-linear least squares techniques defined in Eq. (22).

$$\beta' = E^* O = E^T (EE^T)^{-1} O \tag{22}$$

where, E∗: inverse of E known as Moore-Penrose generalized inverse.

$$\beta' = E^T (\frac{1}{C} EE^T)^{-1} O \tag{23}$$

$$f_L(v) = l(v)\beta' = l(v)\, E^T (\frac{1}{C} EE^T)^{-1} O \tag{24}$$

where, *l(v)* stands for input feature, β is the output weight vector that is calculated using the Moore-Penrose generalized inverse theorem, represented by E^T, C is a constant, and O denotes the target (or label) vector.

$$Y' = Softmax(S) \tag{25}$$

$$Loss = (\frac{1}{K}) \sum_{i=1}^{K} (Y(i) * Log\, Y' + \eta ||\theta||^2 \tag{26}$$

where, K is the dimensional capsule feature length, η is the regularization co-efficient and $|\theta|$ is the constant.

3.6 Walrus evoked extreme feed forward networks

The recommended scheme aims to reduce complexity further. Before training the model, hyperparameter adjustments are made. The hyperparameters that require tuning include the number of hidden layers, hidden units, epochs, and batch size. In this research, the WaOA is used to fine-tune the network attributes for superior classification.

4. IMPLEMENTATION DETAILS

The recommended scheme is implemented using the comprehensive open-source scikit-learn Python framework. The multi-classification algorithm was trained over 100 iterations. The WaOA approach was applied to optimize the loss functions within the networks, resulting in the minimal loss across the iterations.

4.1 Performance metrics

In this section, we highlight the advantages of the recommended scheme in comparison to various DL schemes. To evaluate the effectiveness of the proposed approach, we calculate specific performance metrics. Table 4 provides the quantitative definitions for the metrics used to assess the performance of the recommended scheme.

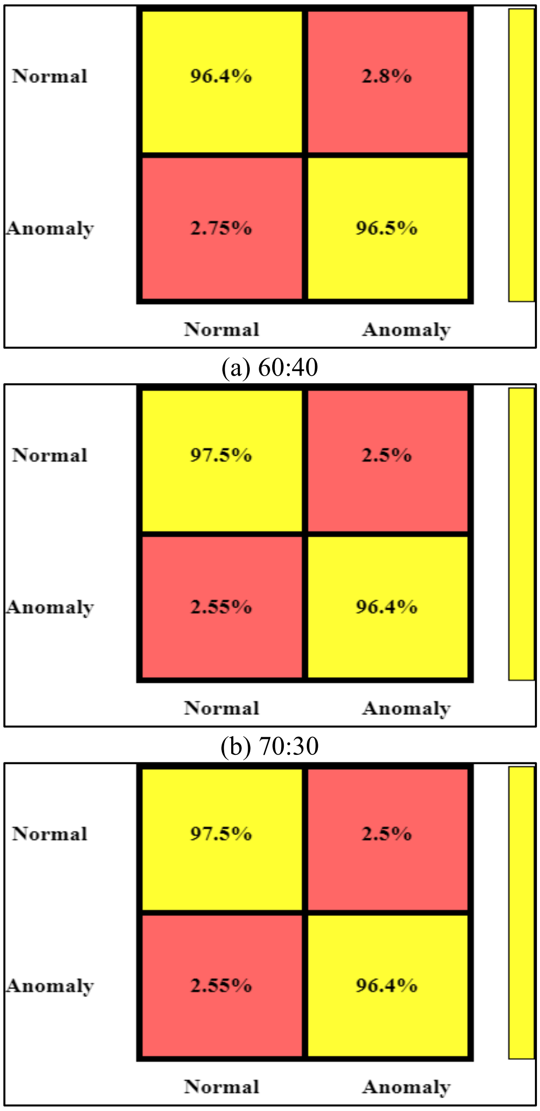
Table 4. Performance measures utilized in the evaluation

Evaluation Metrics	Mathematical Representations
Accuracy	$\frac{TP + TN}{TP + TN + FP + FN}$
Sensitivity or recall	$\frac{\mathrm{TP}}{\mathrm{TP+FN}}$ x100
Specificity	$\frac{TN}{TN + FP}$
Precision	$\frac{TN}{TP + FP}$
F1-Score	$2.\frac{Precison * Recall}{Precision + Recall}$

TP represents True Positive values, TN stands for True Negative values, FP denotes False Positive values, and FN refers to False Negative values.

5. RESULTS AND DISCUSSION

Figure 7 portrays the confusion matrix of the recommended scheme for identifying both anomaly and normal attacks across various ratios of training and testing data. Figure 7 (a) depicts the confusion matrix of the recommended scheme using 60% of the data for training and 40% for testing. Figures 7 (b) and 7 (c) present the confusion matrices of the recommended scheme for 70% of the data used for training and 30% for testing, along with 80% for training and 20% for testing, respectively. From Figure 7, it is evident that the recommended scheme achieved a 96.5% detection rate for anomaly detection.



(a) 60:40

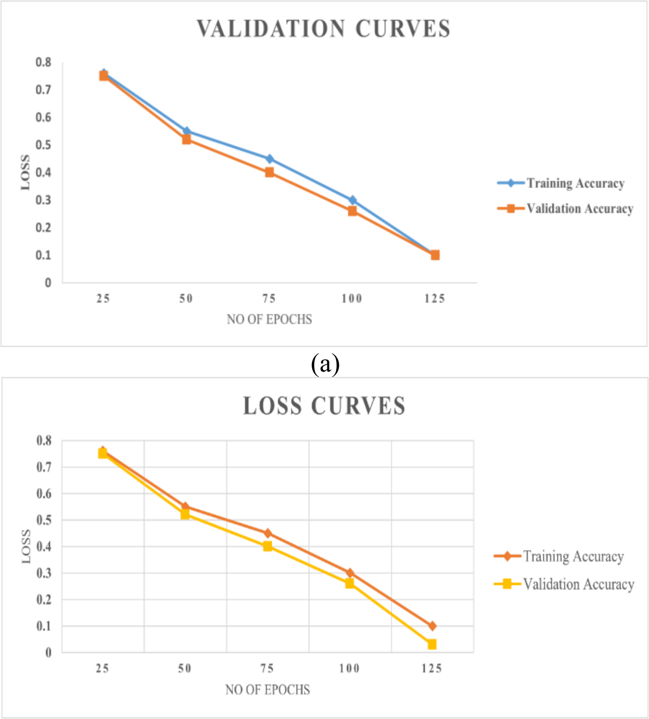
(b) 70:30

(c) 80:20

Figure 7. Confusion matrix of the recommended scheme in detecting the normal and anomaly data

Figure 8 illustrates the validation function used to evaluate the model's effectiveness, with the primary objective being to minimize this loss. It is standard practice to display two distinct curves: one representing the model's performance on the training dataset and the other showing its performance on the validation dataset.

Tables 5 and 6 showcase the benchmarking performance of various learning models in identifying normal conditions and anomalies in the Digital Forensic (DF) environment. Table 5 highlights the evaluation metrics for detecting normal conditions, where the proposed model achieves the highest accuracy of 96%, significantly outperforming other algorithms. Though ELM and ANN have produced the better performance among the other ML models, it can't able to match the effectiveness of the recommended scheme. Similarly, Table 6 presents the results for anomaly detection, where the proposed model again leads with 96% accuracy and outstanding precision and recall values, indicating its robustness in handling diverse scenarios.



(a)

(b)

Figure 8. Validation-loss curve performance of the recommended scheme in detecting both normal and anomalies

Table 5. Benchmarking examination of the varied schemes in identifying the normal conditions in DF environment

Algorithm	Evaluation Measures				
	Accuracy	Precision	Recall	Specificity	F1-Score
SVM	78	77.6	77.6	0.23	77.6
RF	78.4	76.5	76.4	0.24	76.35
DT	75.5	74.3	74	0.26	74.4
ANN	68.5	67.5	67.3	0.334	67.4
ELM	87.4	86.5	86.4	0.114	86.4
Proposed Model	**96**	**95.7**	**95.8**	**0.001**	**96**

Table 6. Evaluation metrics of the varied schemes in identifying the anomalies in DF environment

Algorithm	Evaluation Measures				
	Accuracy	Precision	Recall	Specificity	F1-Score
SVM	78	77.6	77.6	0.23	77.6
RF	78.4	76.5	76.4	0.24	76.35
DT	75.5	74.3	74	0.26	74.4
ANN	68.5	67.5	67.3	0.334	67.4
ELM	87.4	86.5	86.4	0.114	86.4
Proposed Model	**96**	**95.7**	**95.8**	**0.001**	**96**

Figures 9 and 10 illustrate the benchmarking results of the learning models under two distinct scenarios: detecting normal conditions and anomalies in the DF environment. Figure 9 highlights the models' performance in normal condition detection, with the proposed model emerging as the clear leader, achieving a substantially higher accuracy and F1-score compared to other models. Similarly, Figure 10 emphasizes the anomaly detection scenario, where the proposed model once again outperforms all others, showcasing its robustness and precision in identifying anomalies. From the results, it is clear that the WOA tuned Feed forward networks has played the significant role in detecting the anomalies attacks in the DF Environment.

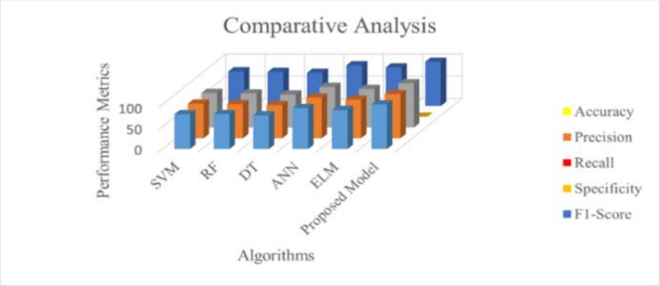


Figure 9. Benchmarking examination of the varied learning models in normal scenario

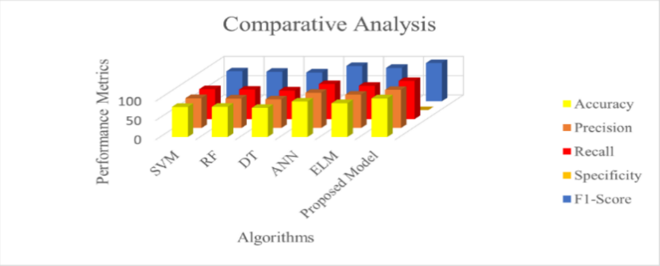


Figure 10. Benchmarking examination of the varied learning models in anomaly detection

6. CONCLUSION

This study presents a forensic investigation framework for anomaly identification using an extreme feedforward learning system integrated with an enhanced walrus-inspired optimization algorithm. The proposed approach effectively addresses the challenges in digital forensic anomaly detection by incorporating centrality-based feature extraction and a robust optimization mechanism. Experimental results demonstrate that the model achieves superior performance with an accuracy of 96%, precision of 95.7%, recall of 95.8%, and an F1-score of 96%, outperforming existing methodologies. These findings highlight the potential of hybrid AI-driven approaches for forensic data analysis, ensuring high detection accuracy and efficiency. Future research can focus on reducing the computational complexity of the proposed framework while enhancing its adaptability to real-time forensic applications. Additionally, integrating advanced DL techniques and expanding dataset diversity could further improve anomaly detection accuracy in evolving digital forensic environments.

REFERENCE

[1] Jarrett, A., Choo, K.K.R. (2021). The impact of automation and artificial intelligence on digital forensics. Wiley Interdisciplinary Reviews: Forensic Science, 3(6): e1418. https://doi.org/10.1002/wfs2.1418

[2] Du, X., Scanlon, M. (2019). Methodology for the automated metadata-based classification of incriminating digital forensic artefacts. In Proceedings of the 14th International Conference on Availability, Reliability and Security, pp. 1-8. https://doi.org/10.1145/3339252.3340517

[3] Krivchenkov, A., Misnevs, B., Pavlyuk, D. (2019). Intelligent methods in digital forensics: state of the art. In Reliability and Statistics in Transportation and Communication: Selected Papers from the 18th International Conference on Reliability and Statistics in Transportation and Communication, RelStat'18, Riga, Latvia. Springer International Publishing. Springer, Cham, 18: 274-284. https://doi.org/10.1007/978-3-030-12450-2_26

[4] Babun, L., Sikder, A.K., Acar, A., Uluagac, A.S. (2022). The truth shall set thee free: Enabling practical forensic capabilities in Smart Environments. In Proceedings of the 2022 Network and Distributed System Security Symposium, San Diego, CA, USA, pp. 24-28. https://doi.org/10.14722/ndss.2022.24133

[5] Shakeel, P.M., Baskar, S., Fouad, H., Manogaran, G., Saravanan, V., Montenegro-Marin, C.E. (2021). Internet of things forensic data analysis using machine learning to identify roots of data scavenging. Future Generation Computer Systems, 115: 756-768. https://doi.org/10.1016/j.future.2020.10.001

[6] Adam, I.Y., Varol, C. (2020). Intelligence in digital forensics process. In 2020 8th International Symposium on Digital Forensics and Security (ISDFS), Beirut, Lebanon, pp. 1-6. https://doi.org/10.1109/ISDFS49300.2020.9116442

[7] Ngejane, C.H., Eloff, J.H., Sefara, T.J., Marivate, V.N. (2021). Digital forensics supported by machine learning for the detection of online sexual predatory chats. Forensic Science International: Digital Investigation, 36: 301109. https://doi.org/10.1016/j.fsidi.2021.301109

[8] Madhukar Rao, G., Ramesh, D. (2020). A hybrid and improved isolation forest algorithm for anomaly detection. In Proceedings of International Conference on Recent Trends in Machine Learning, IoT, Smart Cities and Applications: ICMISC 2020. Singapore: Springer Singapore, pp. 589-598. https://doi.org/10.1007/978-981-15-7234-0_55

[9] Madhukar Rao, G., Ramesh, D. (2020). Ranger random forest-based efficient ensemble learning approach for detecting malicious URLs. In Proceedings of International Conference on Recent Trends in Machine Learning, IoT, Smart Cities and Applications: ICMISC 2020. Singapore: Springer Singapore, pp. 599-608. https://doi.org/10.1007/978-981-15-7234-0_56

[10] Koroniotis, N., Moustafa, N., Slay, J. (2022). A new intelligent satellite deep learning network forensic framework for smart satellite networks. Computers and Electrical Engineering, 99: 107745. https://doi.org/10.1016/j.compeleceng.2022.107745

[11] Palmese, F., Redondi, A.E., Cesana, M. (2022). Feature-sniffer: Enabling IoT forensics in openwrt based wi-fi access points. In 2022 IEEE 8th World Forum on Internet of Things (WF-IoT), Yokohama, Japan, pp. 1-6. https://doi.org/10.1109/WF-IoT54382.2022.10152146

[12] Kalpana, P., Anandan, R. (2023). A capsule attention network for plant disease classification. Traitement du Signal, 40(5): 2051-2062. https://doi.org/10.18280/ts.400523

[13] Wang, S., Jiang, R., Wang, Z., Zhou, Y. (2024). Deep learning-based anomaly detection and log analysis for computer networks. arXiv Preprint arXiv: 2407.05639. https://doi.org/10.48550/arXiv.2407.05639

[14] Nkashama, D.J.K., Félicien, J.M., Soltani, A., Verdier, J.C., Tardif, P.M., Frappier, M., Kabanza, F. (2024). Deep learning for network anomaly detection under data contamination: Evaluating robustness and mitigating performance degradation. arXiv Preprint arXiv: 2407.08838. https://doi.org/10.48550/arXiv.2407.08838

[15] Islam, U., Alwageed, H.S., Farooq, M.M.U., Khan, I., Awwad, F.A., Ali, I., Abonazel, M.R. (2023). Investigating the effectiveness of novel support vector neural network for anomaly detection in digital forensics data. Sensors, 23(12): 5626. https://doi.org/10.3390/s23125626

[16] Yang, L., Moubayed, A., Shami, A., Boukhtouta, A., Heidari, P., Preda, S., Brunner, R., Migault, D., Larabi, A. (2023). Forensic data analytics for anomaly detection in evolving networks. In Innovations in Digital Forensics, pp. 99-137. https://doi.org/10.1142/9789811273209_0004

[17] Jones, G.M., Winster, S.G., Valarmathie, P. (2022). An advanced integrated approach in Mobile Forensic Investigation. Intelligent Automation & Soft Computing, 33(1). http://doi.org/10.32604/iasc.2022.022972

[18] Ashraf, N., Mahmood, D., Obaidat, M.A., Ahmed, G., Akhunzada, A. (2022). Criminal behavior identification using social media forensics. Electronics, 11(19): 3162. https://doi.org/10.3390/electronics11193162

[19] Wei, Y., Chow, K.P., Yiu, S.M. (2021). Insider threat prediction based on unsupervised anomaly detection scheme for proactive forensic investigation. Forensic Science International: Digital Investigation, 38: 301126. https://doi.org/10.1016/j.fsidi.2021.301126

[20] Studiawan, H., Sohel, F. (2021). Anomaly detection in a forensic timeline with deep autoencoders. Journal of Information Security and Applications, 63: 103002. https://doi.org/10.1016/j.jisa.2021.103002

[21] Li, X., Chen, P., Jing, L., He, Z., Yu, G. (2020). Swisslog: Robust and unified deep learning based log anomaly detection for diverse faults. In 2020 IEEE 31st International Symposium on Software Reliability Engineering (ISSRE), Coimbra, Portugal, pp. 92-103. https://doi.org/10.1109/ISSRE5003.2020.00018

[22] Xiao, L., Wan, X., Lu, X., Zhang, Y., Wu, D. (2018). IoT security techniques based on machine learning: How do IoT devices use AI to enhance security? IEEE Signal Processing Magazine, 35(5): 41-49. https://doi.org/10.1109/MSP.2018.2825478

[23] Kalpana, P., Anandan, R., Hussien, A.G., Migdady, H., Abualigah, L. (2024). Plant disease recognition using residual convolutional enlightened Swin transformer networks. Scientific Reports, 14(1): 8660. https://doi.org/10.1038/s41598-024-56393-8

[24] Kalpana, P., Srilatha, P., Krishna, G.S., Alkhayyat, A., Mazumder, D. (2024). Denial of Service (DoS) attack detection using feed forward neural network in Cloud Environment. In 2024 International Conference on Data Science and Network Security (ICDSNS), Tiptur, India, pp. 1-4. https://doi.org/10.1109/ICDSNS62112.2024.10691181

[25] Verma, A., Ranga, V. (2020). Machine learning based intrusion detection systems for IoT applications. Wireless Personal Communications, 111(4): 2287-2310. https://doi.org/10.1007/s11277-019-06986-8

[26] Misra, S., Krishna, P.V., Agarwal, H., Saxena, A., Obaidat, M.S. (2011). A learning automata based solution for preventing distributed denial of service in internet of things. In 2011 International Conference on Internet of Things and 4th International Conference on Cyber, Physical and Social Computing, Dalian, China, pp. 114-122. https://doi.org/10.1109/iThings/CPSCom.2011.84

[27] Kasinathan, P., Costamagna, G., Khaleel, H., Pastrone, C., Spirito, M.A. (2013). An IDS framework for internet of things empowered by 6LoWPAN. In Proceedings of the 2013 ACM SIGSAC Conference on Computer & Communications Security, New York, NY: ACM, pp. 1337-1340. https://doi.org/10.1145/2508859.2512494

[28] Kalpana, P., Malleboina, K., Nikhitha, M., Saikiran, P., Kumar, S.N. (2024). Predicting cyberbullying on social media in the big data era using machine learning algorithm. In 2024 International Conference on Data Science and Network Security (ICDSNS), Tiptur, India, pp. 1-7. https://doi.org/10.1109/ICDSNS62112.2024.10691297

[29] Nabi, S.A., Kalpana, P., Chandra, N.S., Smitha, L., Naresh, K., Ezugwu, A.E., Abualigah, L. (2024). Distributed private preserving learning based chaotic encryption framework for cognitive healthcare IoT systems. Informatics in Medicine Unlocked, 49: 101547. https://doi.org/10.1016/j.imu.2024.101547

[30] Ahmed, N., Rafiq, J.I., Islam, M.R. (2020). Enhanced human activity recognition based on smartphone sensor data using hybrid feature selection model. Sensors, 20(1): 317. https://doi.org/10.3390/s20010317

[31] Trojovský, P., Dehghani, M. (2023). A new bio-Inspired metaheuristic algorithm for solving optimization problems based on walruses behavior. Scientific Reports, 13(1): 8775. https://doi.org/10.1038/s41598-023-35863-5

[32] Attal, F., Mohammed, S., Dedabrishvili, M., Chamroukhi, F., Oukhellou, L., Amirat, Y. (2015). Physical human activity recognition using wearable sensors. Sensors, 15(12): 31314-31338. https://doi.org/10.3390/s151229858

[33] Huang, G.-B., Zhu, Q.-Y., & Siew, C.-K. (2006). Extreme learning machine: Theory and applications. Neurocomputing, 70(1-3), 489-501. https://doi.org/10.1016/j.neucom.2005.12.126