Scientific Reports

https://doi.org/10.1038/s41598-026-37802-6

Article in Press

Artificial intelligence driven approach for securing backup data and enhancing cyber resilience in sustainable smart infrastructure

Received: 24 October 2025

Accepted: 27 January 2026

Published online: 16 March 2026

Cite this article as: Kumar B., Gupta S.K., Dwivedi R. *et al.* Artificial intelligence driven approach for securing backup data and enhancing cyber resilience in sustainable smart infrastructure. *Sci Rep* (2026). https://doi.org/10.1038/s41598-026-37802-6

Basant Kumar, Shashi Kant Gupta, Rashmi Dwivedi, Deema Mohammed Alsekai, Diaa Salama AbdElminaam & Ozlem Kilickaya

ARTIFICIAL INTELLIGENCE DRIVEN APPROACH FOR SECURING BACKUP DATA AND ENHANCING CYBER RESILIENCE IN SUSTAINABLE SMART INFRASTRUCTURE

Basant Kumar[1,2], Shashi Kant Gupta[3,4], Rashmi Dwivedi[5,6], Deema Mohammed Alsekai[7](corresponding author)*

, Diaa Salama AbdElminaam[8,9](corresponding author), Ozlem Kilickaya[10]*

[1]Lincoln University College, Petaling Jaya, Selangor 47301, Malaysia; [2]Modern College of Business and Science,Muscat,Oman; [3]Lincoln University College, Petaling Jaya, Selangor 47301, Malaysia;[4]Chitkara University Institute of Engineering and Technology, Rajpura, Punjab 140401, India; [5]Muscat University, Muscat,Oman;[6]Alliance University,Bangalore,India; [7]Princess Nourah Bint Abdulrahman University, P.O. Box 84428, Riyadh 11671, Saudi Arabia; [8]Faculty of Computers and Artificial Intelligence, Benha University, Benha, Egypt; [9]Jadara Research Center, Jadara University, Irbid, 21110, Jordan; [10]University of the People, Pasadena,, USA

pdf.basantkumar@lincoln.edu.my; basant@mcbs.edu.om

raj2008enator@gmail.com;rdwivedi@muscatuniversity.edu.om; dmalsekait@pnu.edu.sa; diaa.salama@miuegypt.edu.eg; ozlem.kilickaya@uopeople.edu

ABSTRACT

A crucial factor for smart cities, which are more vulnerable to cyber threats, is Cyber Resilience (CR). Nevertheless, the conventional frameworks didn't concentrate on assuring the Backup Data (BD) integrity before restoration, showing less resilience. Therefore, this article implements an AI-powered BD integrity verification approach for CR in smart infrastructure using Murmur Polytopes Hash (MPH). Initially, the nodes are initialized in the smart city, followed by node clustering, data security, and storage (cloud server and Interplanetary File System (IPFS) (backup)). Now, the hash code is generated and updated in the Merkle tree. Besides, to perform data collection, pre-processing, clustering, correlation heatmap generation, feature extraction, and attack classification, the proposed ransomware attack detection module is designed. If the data is attacked, then the BD verification is done using MPH. Then, the BD is restored. If the data is normal, then the data is downloaded from the cloud server. Thus, the proposed work had a high security level and accuracy of 98.45% and 98.65%, respectively, showing better resilience.

Keywords: *Cyber-Security (CS), Artificial Intelligence (AI), Cyber Resilience (CR), Green Smart Infrastructure (GSI), Backup Data Integrity (BDI), Ransomware Attack (RA), and Energy Efficiency (EE).*

1. INTRODUCTION

Recently, smart cities have played a vital role in augmenting the citizens' quality of life [1]. Potential security issues have also increased with the rapid advancement of Internet of Things (IoT)-assisted smart infrastructures [2]. Yet, RA is referred to as the most dangerous type of malware attack [3]. Thus, AI has emerged to reduce the significant impact caused by malware attacks [4, 5]. AI plays a fundamental role in numerous applications owing to its beneficial

natures, such as operational efficiency, innovation, and decision-making [6]. An automatic and robust AI-assisted cybersecurity framework enables systems to detect and mitigate cyber threats without human intervention [7].

The existing studies implemented AI techniques like Naïve Bayes (NB), k-nearest neighbour, and Long Short-Term Memory (LSTM) for detecting RA [8]. However, increased energy consumption is caused by AI-driven threat detection and large-scale data computation [9, 10]. Thus, a green cybersecurity framework based on lightweight attack detection modules was established in [11, 12]. Yet, none of the prior works concentrated on BDI verification before restoration, affecting the CR. Therefore, a BDI-aware CR framework for GSI is essential. Thus, this research is motivated by the need to provide an enhanced AI-assisted BDI verification framework using MPH for CR in GSI.

1.1 Problem statement

The prior frameworks' certain drawbacks are listed below,

- None of the traditional studies focused on verifying the backup data before restoration, degrading the CR.
- The conventional [13] failed to protect the sensitive information during data transmission, thus resulting in data breaches.
- The prevailing work [14] struggled to handle the dynamic and evolving threat patterns, showing less adaptability.
- The existing study [15] didn't ensure an energy-efficient cybersecurity framework for smart cities, thereby depicting poor environmental sustainability.

1.2 Objectives

The proposed methodology's major goals are stated further,

- A robust MPH is employed to evaluate the integrity of the backup data before restoration, increasing the CR.
- The proposed Cocks-Pinch Curve Cryptography (CPCC) proficiently protects sensitive information during data transmission.
- By incorporating Transfer Learning (TL), the proposed TL-SSGGSRU adaptively handles the evolving threat patterns.
- The proposed Parzen Bhattacharyya Chernoff-based K-Means (PBCK-Means) is established to group the nodes, thus ensuring energy efficiency.

The paper is organized as: the associated studies are reviewed in Section 2, the proposed methodology is demonstrated in Section 3, the proposed work's performance is appraised in Section 4, and Section 5 concludes the paper.

2. RELATED SURVEY

Islam et al. [15] presented a dynamic cybersecurity risk assessment scheme with interpretability for enhanced resilience of digital infrastructure. To identify the cyber threats, a linear regression and Deep Learning (DL) were established. This model had high trustworthiness. Yet, this work didn't ensure EE in cybersecurity.

Alazab et al. [14] presented an improved threat intelligence scheme for better CR. This work utilized ensemble classifiers, including Random Forest (RF), NB, along with logistic regression, to identify the real-time threats. This approach had a high detection rate. Nevertheless, this model struggled to capture the unknown threat patterns.

Ali et al. [13] displayed an enhanced AI-powered cybersecurity risk management scheme and resilience of the national critical infrastructure. The different classifiers, such as NB, Convolutional Neural Network (CNN), and Recurrent Neural Network (RNN), were utilized to identify the cyber-threats. This model promoted robust defence strategies. Nevertheless, this framework failed to mitigate the risk of sensitive information leakage.

Zhang et al. [16] presented an unknown encryption ransomware attack detection framework centered on dual generative adversarial networks. This model had high adaptability and robustness. Yet, this scheme had a mode collapse issue.

Singh et al. [17] recognised an enhanced RA detection based on TL along with DL ensemble models using cloud-encrypted data. To perform RA classification, a CNN with pre-trained transformers was introduced. This model ensured high cloud security. However, this scheme had considerable computational complexity.

Nandanwar & Katarya [18] propounded a robust and privacy-aware intrusion detection system for IoT networks centered on hybrid blockchain and Federated Learning (FL). This work utilized elliptic curve cryptography with FL algorithms to ensure secure network operations. Here, a CNN-bidirectional LSTM was employed to detect the intruder behaviours in the IoT network. This approach ensured decentralized and privacy-preserving threat detection across IoT devices. But, this framework failed to mitigate the impact of gradient model leakage, degrading the network resilience.

Yazdinejad et al. [19] demonstrated an interpretable and privacy-preserving FL scheme for threat detection in cyber-physical-social systems. Here, differential privacy and partially homomorphic encryption were utilized to ensure secure gradient model updates. Thus, the suggested FL scheme enabled privacy-aware and secure threat detection. Also, the presence of SHapley Additive exPlanations (SHAP) aided in enhancing the model's trustworthiness in network threat detection. Nevertheless, this methodology struggled to handle non-independent and identically distributed data across heterogeneous clients.

Yazdinejad et al. [20] implemented a resilient privacy-preserving FL approach for model poisoning attack mitigation. The Gaussian mixture model and Mahalanobis distance for Byzantine-tolerant aggregation were established to perform secure threat detection. To ensure confidentiality, additive homomorphic encryption was established. This approach had superior performance in intrusion detection. Yet, it had scalability bottleneck issues, degrading the model's trustworthiness and robustness.

3. PROPOSED AI-POWERED CR FRAMEWORK FOR GSI

The proposed work implements an improved AI-assisted backup data integrity verification approach using MPH for CR in sustainable smart infrastructure. In Figure 1, the proposed work's architecture is given.

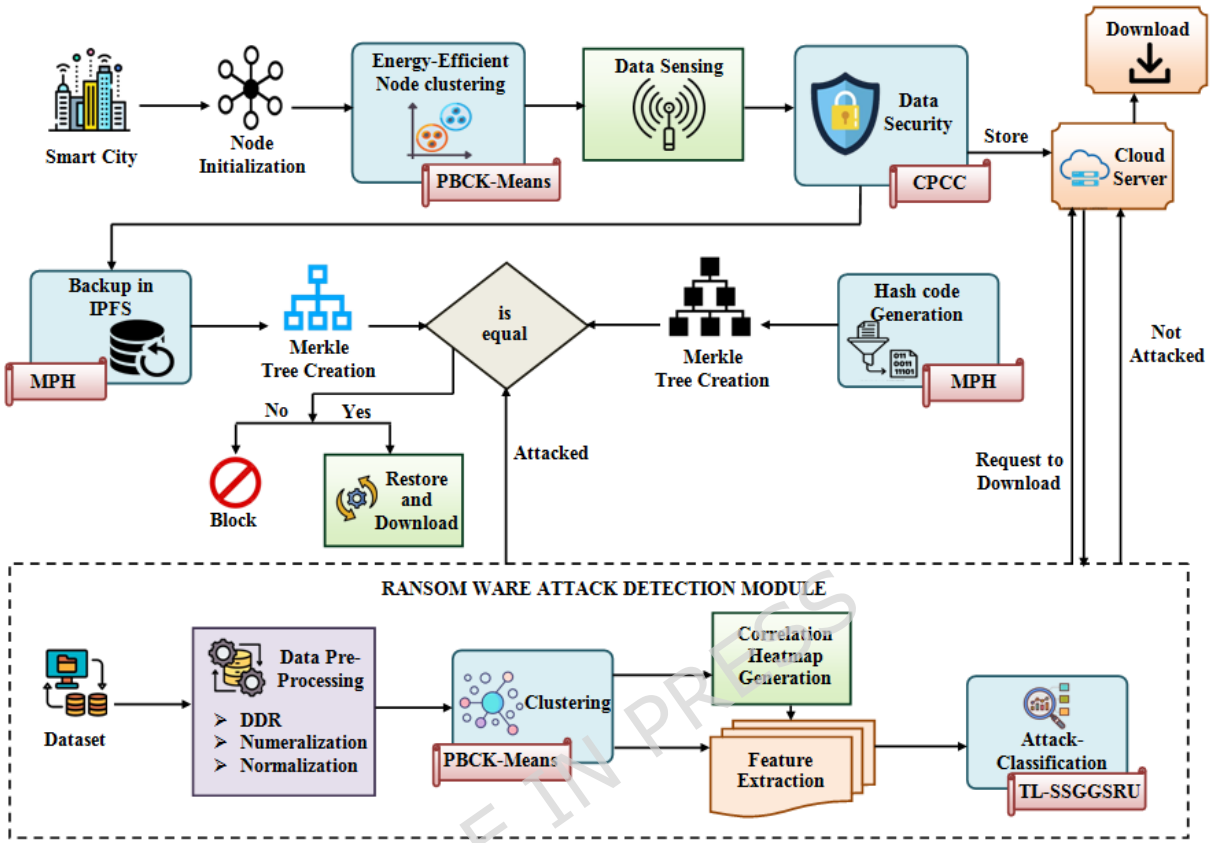

Figure 1: Diagrammatic design of the research methodology

3.1 Node initialization

Initially, the sensor nodes deployed across the smart city are initialized by considering the network addresses and IDs.

$$d_y = \langle d_1, d_2, \text{K K } d_Y \rangle \text{ Where, } y = 1 to Y \quad (1)$$

Where, Y demonstrates the number of initialized nodes $\square_y$.

3.2 Clustering

Next, $\square_y$ is clustered based on the minimum distance using the proposed PBCK-Means, ensuring an energy-efficient smart city. The K-Means ensures that all the nodes are uniformly distributed within each group, ensuring high efficiency. But, the K-Means struggled to handle the clusters with varying density. Hence, the Bhattacharyya Chernoff Distance (BCD) is used to allow clusters of different densities by measuring the statistical similarity between probability distributions. Also, the K-Means had suboptimal outcomes. Therefore, the Parzen Window Initialization (PWI) is introduced to initialize the centroid points. Firstly, the centroid point $\square$ is selected from the available data points $\square_y$ using PWI.

$$\square_y \square \frac{1}{y\square} \sum_{y\square 1}^{Y} \square \left(\frac{\square_1 \square \square_y}{\square} \right) \tag{2}$$

Here, $\square$ designates the window width, $\square$ depicts the data dimensions, and $\square$ illustrates the kernel function. Next, the distance $\square[\square]$ betwixt the data point and the centroid is calculated using BCD.

$$\square[\square,\square] \square \ln\left[\sum_{y} \square_y \square \left(\frac{\square}{\square_y} \right)^{\square\square} \right] \qquad \square \square [0,1[\tag{3}$$

Here, $\square[\square,\square]$ indicate the probability distributions of $\square_y$, ln depicts the natural logarithm function, and $\square$ exhibits the tuning parameter. Afterwards, the data points are allocated to the cluster, which has the closest distance. Here, the clustered nodes are mentioned as $\square[\square]$.

3.3 Data sensing

Afterwards, the $\square$ collect sensor data and then forwards it to the cloud server.

$$\square_i \square \square^{\text{sensor data}} \square \square[\square_1, \square_2, \square \square \square_I[\text{ Here, } i \square 1 to I \tag{4}$$

Next, the I number of sensed data $\square_i$ is fed into the data security.

3.4 Data security

The proposed CPCC effectively encrypts $\square_i$ to mitigate the risk of sensitive information leakage. The Elliptic Curve Cryptography (ECC) provides better security. Nevertheless, the ECC had high computational overhead. Hence, the Cocks-Pinch Curve (CPC) $\square[\square]$ is established to minimize the impact of negative points, thus reducing the computational overhead. The proposed CPCC uses a cocks-pinch elliptic curve that aids in increasing the attack resistance level. A generator point computed from the curve is used to create encryption keys. Here, the sizes of the private key and public key are 512 bits and 1024 bits, respectively. The high key sizes ensure a better security level. The presence of CPC aids in generating the keys with high randomness. Further, the encryption keys are stored in a compressed form to save memory.

$$\square : \square^2 \square\square^3 \square\square\square \square\square\square \text{mod} \square[\tag{5}$$

Here, $\square$ displays the finite field of integers, $\square[\square,\square[$ denote the curve parameters, mod depicts the modulo operation, $\square[\square,\square[$ demonstrate the coefficients chosen from $\square$, and $\square$ illustrates the prime number. Similarly, the private key $\square[\square]$ is randomly selected, and then, the public key $\square[\square$ is created by multiplying the private key and curve point $\square[\square]$.

$$\square \square\square\square\square[\square] \tag{6}$$

Afterward, the encryption is done using two-cipher components $\square[\square_1, \square_2[$.

$$\square_1 \square\square\square\square h \tag{7}$$

$$\square_2 \square\square_i \square\square\square\square \tag{8}$$

Here, $\square$ identifies the random session key.

Contrarily, the data is decrypted by considering the private key.

$$\square_i \square\square_2 \square\square\square\square_1 \tag{9}$$

Also, the $j \square 1 to J$ number of encrypted data $\square_j$ is stored in the cloud server for the purpose of future resource access. Before storing the data in the cloud server, the hash value is computed, and then, the Merkle tree is constructed.

3.5 Hash code generation

Here, by using the proposed MPH, the hash value of $\square_j$ is generated for backup data integrity verification. The Murmur Hash (MH) had faster computations and minimum hash collisions. But, the MH selects the initial seed value in a random manner. Therefore, the proposed approach introduces the Polytopes Formula (PF) to initialize the seed value with high randomness, showing a high security level. Firstly, a seed value $\square\square$ is initialized to compute the initial hash value.

$$\square\square \frac{\sqrt{j \square 1}}{j! \sqrt{2^j}} \square\square \square_j \tag{10}$$

Here, $\square$ depicts the scaling factor. The input data $\square_j$ is processed in fixed-size blocks.

$$\square_j \square \sum_{u \square 1}^{U} \left| \square_u \right| \tag{11}$$

Where, U specifies the number of blocks. Each block $\left| \square_u \right|$ is multiplied by specific constants. Next, the steps like left rotation, XOR operation, and mixing operations are carried out.

$$\square_u^{'} \square\square_u \square b_1 \tag{12}$$

$$\square_u^{\text{rotation}} \square\square\square\square_u^{'}, 15\square \tag{13}$$

$$\square_u^{''} \square\square_u^{\text{rotation}} \square b_2 \tag{14}$$

Here, $\square_u^{'}$, $\square_u^{\text{rotation}}$, and $\square_u^{''}$ signify the updated block after constant multiplication, left rotation, and another constant multiplication, respectively, $\square b_1, b_2 \square$ indicate the constant values, and $\square$ demonstrates the left shift operation.

$$\square \square\square \text{xor} \square_u^{''} \tag{15}$$

$$\square^{\text{rotation}} \square \square \square,13[\tag{16}$$

$$\square \square \square^{\text{rotation}} \square 5 \square b_1 \tag{17}$$

Where, $\square$, $\square^{\text{rotation}}$, and $\square$ display the updated hash value after the XOR operation, left rotation, and mixing operation, correspondingly. After processing all blocks, a final mixing function is applied to compute the hash value $\square[$. The proposed MPH's pseudo code is given below,

---

Input: Encrypted data $\square_j$

Output: Hash value $\square$

Begin

Initialize $\square_j$, $\square$ and $\square$

For 1 to each $\square_j$ do,

#seed initialization

Apply PF $\square\square \frac{\sqrt{j \square 1}}{j!\sqrt{2^j}} \square \square^j$

Determine block size

$$\square_j \square \sum_{u \square 1}^{U} |\square_u|$$

Apply block mixing operations

$$\square_u^{''} \square \square_u^{\text{rotation}} \square b_2$$

$$\square \square \square^{\text{rotation}} \square 5 \square b_1$$

Perform tail handling

Execute final mixing function

Compute final hash value

End For

Return $\square$

End

3.6 Merkle tree creation

Next, to ensure efficient data integrity verification, the Merkle tree is constructed from $\square$. Initially, the leaf nodes are combined together, and then, their hashes are fused. Based on the concatenated hashes, the parent nodes are created. The concatenation process $\square[$ is repeated until a Merkle root $\square^{\text{root}}[$ is obtained.

$$\square^{\text{root}} \square \sum_{l \square 1}^{L} \square_{1,2} \| \square_{l,L} \tag{18}$$

Here, $l \square 1,2,\square L$ specifies the number of hash values. Then, $\square^{\text{root}}$ is updated in the cloud server.

3.7 IPFS

Meanwhile, $\square_j$ is updated in the IPFS (decentralized storage) to serve as a reliable backup in case of attack scenarios. In IPFS, the hash code $\square_{\text{IPFS}}$ is automatically created from $\square_j$ using the proposed MPH and then stored in the Merkle tree format. The processes involved in the hash code generation and Merkle tree creation are already discussed in Sections 3.5 and 3.6, respectively.

Still, $\square_j$ is subjected to the proposed RA detection module.

3.8 Ransomware attack detection

It is a sort of malware that encrypts the user's personal data until a ransom is paid. The proposed RA detection framework is explained below,

3.8.1 Data collection

Firstly, to train the RA prediction module, the historical ransomware attack detection dataset $D\square$ is gathered.

3.8.2 Pre-processing

Next, to augment the data quality, $D\square$ is pre-processed. The redundant information presented in $D\square$ is eliminated. Next, the duplicate data removal is subjected to numeralization, where the non-numeric terms are converted into numerical values. Finally, the numerical vectors $\widetilde{F}$ are normalized within the 0 to 1 range by employing min-max normalization.

$$\hat{\square} = \frac{\widetilde{F} - \min(\widetilde{F})}{\max(\widetilde{F}) - \min(\widetilde{F})} \tag{19}$$

Here, $\hat{\square}$ denotes the pre-processed data.

3.8.3 Clustering

Likewise, $\hat{\square}$ is clustered using the proposed PBCK-Means based on similar patterns to increase the computational efficiency. The steps in the proposed PBCK-Means are explained in Section 3.2.

3.8.4 Correlation heatmap generation

From the clustered data, the correlation heatmap (visual representation) is generated to showcase the relationship among the features. The correlation heatmap $\square_{\square}$ provides detailed insight into how strongly the features are related to each other, enhancing the classification accuracy.

3.8.5 Feature extraction

Here, the data features like file name, hash value, debug size, export size, and resource size are extracted from . Also, the correlation heatmap features, such as correlation, colour intensity, patterns, and diagonal line, are extracted from . The extracted features are fed into the attack classification.

3.8.6 Attack classification

Here, is fed into the proposed TL-SSGGSRU, which classifies the data into attack or normal. The Gated Recurrent Unit (GRU) has less memory requirement and efficient computation. But, the GRU had overfitting issues. Thus, the proposed work deploys the Shake-Shake Regularization (SSR) technique to diminish the over-fitting issues. Moreover, the GRU had poor learning efficiency. Hence, the Swim Activation (SA) function is established to increase learning efficiency. The SA smooths gradient flow and adaptively balances non-linearity. Further, the TL is used to adapt to the evolving threat patterns. In Figure 2, the proposed TL-SSGGSRU's architecture is given.

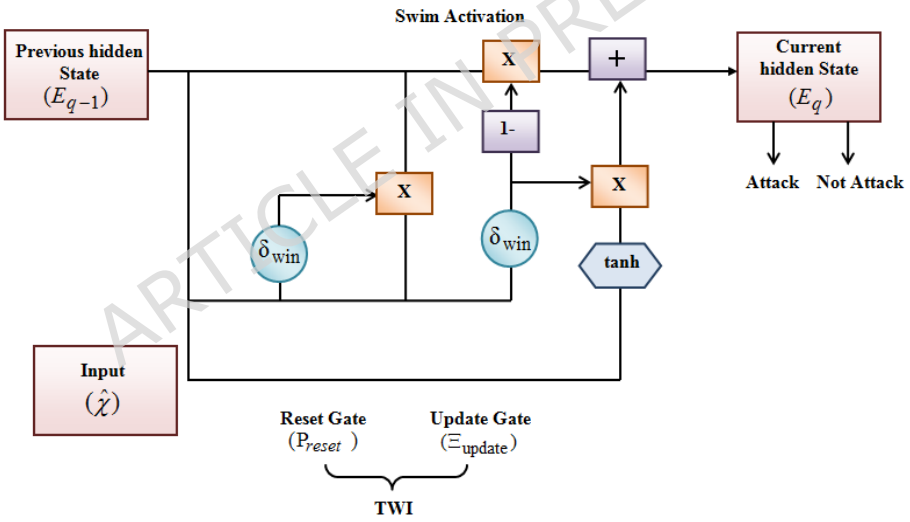

Figure 2: TL-SSGGSRU's design

Firstly, the input and previous hidden state's information are fed into the reset gate that eliminates the inappropriate information as of the past state.

$$\text{reset} = \text{win}[Wt, E_{q-1}] + B^{ias} \quad (20)$$

$$\text{win} = \frac{1}{2}\left(1 + \frac{}{\sqrt{1+^2}}\right) \quad (21)$$

Here, $\square_{win}$ specifies the SA, Wt depicts the weight value, B^{ias} illustrates the bias value, and $\square$ exhibits the trainable parameter. Further, the SSR is used to regularize the weight values.

$$\text{Wt} \square \square \square V_1 \square \square 1 \square \square V_2 \square \quad (22)$$

Where, $\square$ signifies the random coefficient, and V_1, V_2 exhibit the outcomes of the two residual gates. Similarly, the update gate $\square_{update}$ decides the information that needs to be included in the present state.

$$\square_{update} \square \square_{win} \square \text{Wt} \square, E_{q-1} \square \square B^{ias} \quad (23)$$

Later, the candidate's hidden state $\hat{E}_q$ is computed to hold the relevant information for a long period.

$$\hat{E}_q \square \square_{win} \square \text{Wt} \square \square \square \square_{update} \square \square_{reset} \square E_{q-1} \square \square B^{ias} \quad (24)$$

Also, the present hidden state E_q is estimated based on the previous gates.

$$E_q \square \square 1 \square \square \square_{reset} \square \square E_{q-1} \square \square \square_{reset} \square \hat{E}_q \quad (25)$$

Also, TL T_{learn} is enabled to capture the rare threat patterns. TL offers a mechanism to learn from limited data, making it effective for recognizing unknown patterns.

$$T_{learn} \square \min \square \square L_{oss} \square, E_{q \square 1} \square E_q ; \square \square \quad (26)$$

Here, $\square$ denotes the pre-trained model's parameters, and L_{oss} indicates the loss function. Grounded on the learned features from the hidden state, the proposed TL-SSGGSRU classifies the data into attack At or normal Nl.

$$\square \square \text{At}, \text{Nl} \quad (27)$$

Here, $\square$ depicts the prediction outcome. The proposed TL-SSGGSRU's pseudo code is specified below,

---

Input: Extracted features $\square$

Output: Attack classification $\square$

Begin

Initialize $\square$, $\square_{reset}$, Wt, $\square_{update}$ and E_{q-1}

For 1 to each features do,

Execute reset gate

$$\square_{reset} \square \square_{win} \square \text{Wt} \square, E_{q-1} \square \square B^{ias} \square$$

Activate swim function

$$\square_{win} \square \square \square \square \frac{1}{2} \square 1 \square \frac{\square \square}{\sqrt{1 \square \square^2}} \square$$

Regularize weight value

$$Wt = W_1 \cdot 1 + W_2$$

Evaluate update gate

$$\sigma_{update} = \sigma_{win}[Wt, E_{q-1}] + B^{ias}$$

Compute hidden state

$$E_q = (1 - \sigma_{reset}) \cdot E_{q-1} + \sigma_{reset} \cdot \hat{E}_q$$

Integrate TL T_{learn}

Classify attack or normal

End For

Return $\{At, Nl\}$

End

If the data is normal, then the data is securely downloaded from the cloud server. If the data is attacked, then the BDI verification is done for reliable backup.

3.9 Backup data integrity verification

Here, the backup data's integrity is verified by analogizing the hash value stored in the cloud server and the hash value stored in the IPFS with respect to the time stamp. If both the hash values are equal, then the backup data is restored in a reliable manner. Contrarily, the backup data is blocked from the network. The state machine diagram of the restore decisions is shown in Figure 3.

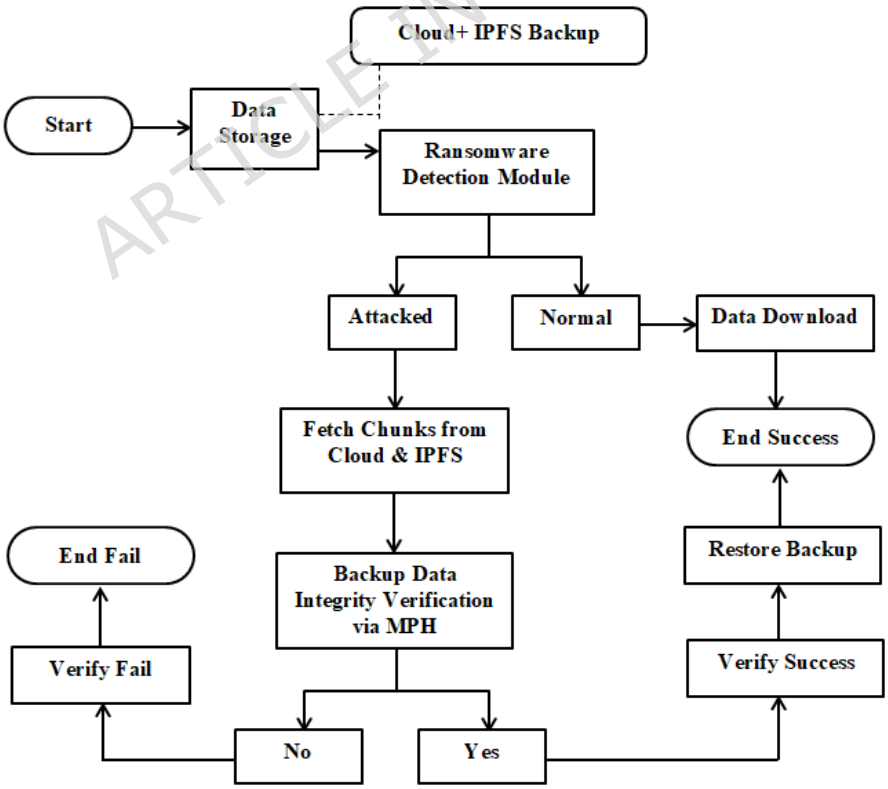

Figure 3: State machine diagram of the restore decision

The proposed work ensures cloud-IPFS synchronization by generating MPH hash code and storing it in the cloud and IPFS in a commit state. Also, the rollback prevention is done by accepting only data whose Merkle root matches the latest MPH snapshot. Older roots get blocked. Cross-root verification is done by validating only if it matches both the cloud and IPFS merkle root from the same state.

The possible threats include older root replay (rollback), malware inserted into the clustering stage (clustering poisoning), and cloud storage tampered. The maps for threat mitigation are discussed below.

To mitigate the rollback threats, the proposed MPH is used to perform backup data integrity verification.

The inclusion of PWI aids in initializing the stronger centroid that mitigates the impact of cluster poisoning.

A novel CPCC-based encryption is employed to encrypt the sensitive information before storing it in the Merkle tree and cloud, thereby minimizing the risk of tampering.

Thus, the proposed methodology provides a robust CR framework with minimum energy consumption.

4. RESULT AND DISCUSSION

In this section, the proposed work's resilience is evaluated through performance assessment and comparative validation. The experimental specifications are presented below,

Experimental setup - The implementation of the proposed methodology is done on the PYTHON platform. Python is a general-purpose and high-level programming language, known for its versatility. PYTHON is widely adopted by many researchers owing to its extensive libraries, scalability, and portability. Also, Python is a dominant language in AI-based approach development. The hardware configurations of the proposed methodology are stated below,

- CPU Speed: 3.20 GHz
- Processor: Intel i5/core i7
- Operating system: Windows 10pro
- RAM: 8 GB
- Hard disk space: 16 GB for 32-bit OS
- Display: 800 x 600
- Graphics card: DirectX 9

Moreover, the experimental parameters of the proposed methodologies are illustrated in Table 1,

Table 1: The parameters of the proposed methodologies

Parameters	Specifications
TL-SSGGSRU	
Input layer	32
Hidden layer	3
Hidden layer neurons	258, 128, 64
Dropout	0.2
Activation	Swim
Optimizer	Adam
Weight regularization	Shake-Shake
Learning rate	0.0001
Batch size	64
Epochs	10
Loss function	Categorical cross-entropy
PBCK-Means	
Number of clusters	5,10,15
Initialization method	Parzen window initialization
Maximum iterations	300
Distance metric	Bhattacharyya-Chernoff
Batch size	1000
Sampling rate	10 repeated runs
Confidence interval	95% over 10 runs

Table 1 exhibits the experimental parameters of the proposed TL-SSGGSRU and PBCK-Means regarding attack classification and energy-efficient node clustering, respectively. Learning rate (0.0001) controls the step size that determines the model weight updates during training. Batch size (64) refers to the number of training samples processed before the model updates its weights. Further, the Adam optimizer is used to adjust model weights to minimize the loss function named categorical cross-entropy. Here, dropout is used to prevent overfitting. An epoch is the number of times the whole training dataset is fed into the proposed model during training. Regarding node clustering, the proposed work considers a batch size of 1000. Batch size in clustering refers to the number of data points processed per iteration.

4.1 Dataset description

To assess the proposed framework, the Ransomware Detection Dataset (RDD) is utilized. The RDD involves the features extracted from Windows Portable Executable files. The dataset samples are shown in Table 2.

Table 2: Dataset characteristics

Class	**Samples**
Normal	27118
Attack	35367
Total	62485

The whole dataset is divided into '2' sets, namely training (80% - 49988) and testing (20% - 12497).

4.2 Performance assessment of the proposed approach

The proposed work's performance is validated by comparing it with several existing techniques with respect to the quality metrics. In the proposed work, the quantitative metrics and qualitative metrics, including accuracy, precision, recall, False Positive Rate (FPR), clustering time, energy consumption, latency, and computational complexity, are measured to assess the proposed schemes' performance. The above-mentioned performance metrics are measured by using the computational formula, which is presented in Table 2.

Table 3: Performance metrics measurement

Metrics	**Formula**
Accuracy	$\frac{T^{\square} \square T_{\square}}{T^{\square} \square T_{\square} \square F^{\square} \square F_{\square}}$
Precision	$\frac{T^{\square}}{T^{\square} \square F^{\square}}$
Recall	$\frac{T^{\square}}{T^{\square} \square F_{\square}}$

Specificity	$\frac{T_{\square}}{T_{\square} \square F^{\square}}$
FPR	$\frac{F^{\square}}{F^{\square} \square T_{\square}}$
Training time	Training end time – start time
Security level	$\frac{1}{\square\square \square\square}$
Energy consumption	$\bar{\square} \square\square_{\square}$
Latency	$L^{\text{prob}} \square L^{\text{Trans}} \square L^{\text{Que}} \square L^{\text{Pro}}$
Collision rate	$\frac{\text{Number of collisions}}{\text{Total transmission attempts}}$
Randomness	$1 \square \sum_{u \square 1}^{U} \text{Pr}_u^{\,2}$
Brier score	$\frac{1}{\square\square} \square \square \text{Pd}_{\square\square} \square \text{Aq}_{\square\square}\square$
Hashcode generation time	Hashcode generation end time – start time
Hashcode creation time	Hashcode creation end time – start time
Encryption time	Encryption end time – start time
Decryption time	Decryption end time – start time
Clustering time	Clustering end time – start time

In Table 3, $\square\square$ denotes the probability of attack, $\square$depicts the impact of attack, $\square$ and$\square_{\square}$ illustrate the power and time, respectively, L^{prob}, L^{Trans}, L^{Que}, and L^{Pro} demonstrate the

propagation delay, transmission delay, queuing delay, and processing delay, respectively, $T_{\square}$, $T_{\square}$, $F_{\square}$, and $F_{\square}$ depict the true positives, true negatives, false positives, and false negatives, respectively, $\square 1 to \square$ exhibits the number of predictions, $\mathrm{Pd}_{\square}$ indicates the predicted probability of event $\square$, $\mathrm{Aq}_{\square}$ demonstrates the actual outcome, $u \square 1 to U$ denotes the total number of possible hash value outputs, and Pr_u demonstrates the probability of u^{th} hash value appearing. Regarding hash generation performance validation, the collision rate indicates the number of collisions that occur over total transmission attempts. Also, randomness analyzes bit sequences to compute entropy. Brier score compares the predicted probabilities to actual outcomes. Energy measures power consumption during transmission/reception over time. Energy is measured using device counters or simulation models. Some devices report energy or power usage using system Application Programming Interfaces (API). Latency measures the time difference between the packet sent and the packet received. In basic, latency is measured via network simulators or Wireshark timestamps. Throughput quantifies the total data transmitted successfully with respect to time. Based on the device /system logs, log events, timestamps, errors, and power states are considered in the proposed work to measure the metrics, including energy, latency, and throughput. Further, the metrics, such as accuracy, precision, and specificity, are measured based on statistical computation using prediction logs.

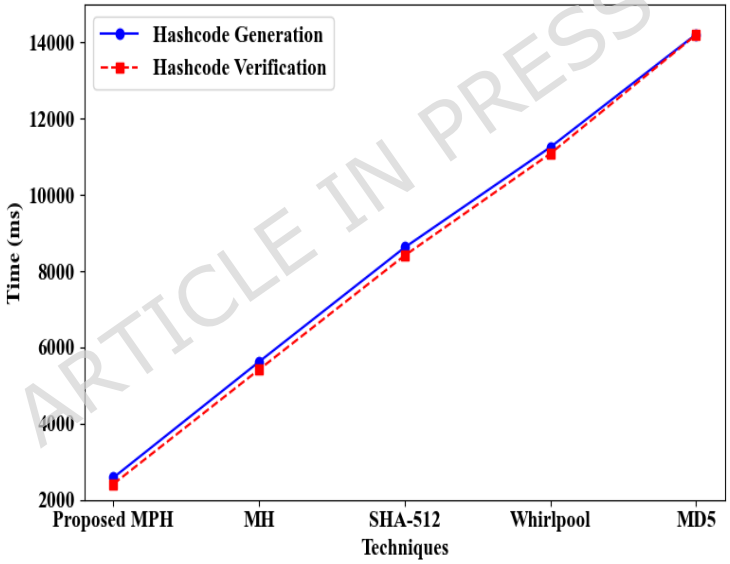

Figure 4: Performance validation for BDI verification

The proposed MPH's performance is weighed against the prevailing MH, Secure Hashing Algorithm-512 (SHA-512), Whirlpool, and Message Digest-5 (MD5) in Figure 4. The proposed MPH consumed 2584ms and 2415ms to complete hashcode generation and hashcode verification, respectively. Nevertheless, the conventional techniques had considerable time consumption due to the lack of proper initial hash value selection. Improper hash value selection causes suboptimal performance. Thus, the MPH had higher supremacy owing to the inclusion of PF. The PF effectively selects the seed by calculating entropy-weighted polytope volume with maximum variance. Therefore, the proposed MPH outperforms existing algorithms by producing faster and unbiased outcomes.

Table 4: Numerical investigation of the proposed MPH

Algorithms	Hashcode generation time (ms)
Proposed MPH	2584
SHA-256/3 with HMAC	3487

Table 4 compares the performance of the proposed MPH and the conventional Secure Hashing Algorithm-256/3 with Hash-based Message Authentication Code (SHA-256/3 with HMAC). The proposed MPH consumed 2584ms to generate a hashcode, whereas the prevailing SHA-256/3 with HMAC obtained a hashcode generation time of 3487ms. However, the existing SHA-256/3 with HMAC heavily relies on a random initial seed, increasing the computational overhead. Improper initial hash value selection increases the iteration process, thus resulting in poor performance. In the proposed MPH, PF generates a well-distributed seed by sampling multi-dimensional polytope vertices, thus reducing overhead issues. Overall, the proposed MPH obtains faster hashcode generation with high randomness.

Table 5: Dieharder test for BDI verification

Algorithms	Average p-value	Standard deviation of p-value	Dieharder verdict (Final decision)
Proposed MPH	0.501	0.072	Pass
MH	0.498	0.080	Pass
SHA-512	0.492	0.095	Pass
Whirlpool	0.381	0.141	Weak
MD5	0.217	0.192	Weak

Table 5 shows the Dieharder test of the proposed MPH and extant techniques regarding BDI verification. Dieharder test is a computer program that checks whether a sequence of numbers of bits looks random enough or not. The dieharder test verifies uniformity, patterns, independence, and entropy, thereby ensuring high security. The proposed MPH obtains an ideal p-value (approximately 0.5) with the least deviation, showing strong randomness. In the proposed MPH, PF creates mathematically derived seed values with high uniformity, ensuring less predictable and more random hash values. Therefore, the proposed work obtained better performance significantly rather than random manner. However, the existing whirlpool had a higher standard deviation with a slight bias, illustrating weak randomness. Thus, the dieharder test analysis proved that the proposed MPH was more resilient than existing approaches in BDI verification.

Table 6: Empirical analysis for BDI verification

Techniques	Collision rate (%)	Randomness (bits)
Proposed MPH	0.0004	124.3
MH	0.0009	167.5
SHA-512	0.0083	189.7
Whirlpool	0.0144	252.2
MD5	0.1492	281.3

Table 6 analyzes the collision rate and randomness of the proposed MPH and existing techniques with respect to the BDI verification. The proposed MPH obtained a collision rate and randomness of 0.0004% and 124.3bits, respectively. However, the conventional MD5 attained a collision rate and randomness of 0.1492% and 281.3bits, correspondingly. Hence, the analysis outcomes exhibited that the proposed method had higher security and resistance in BDI verification than traditional techniques.

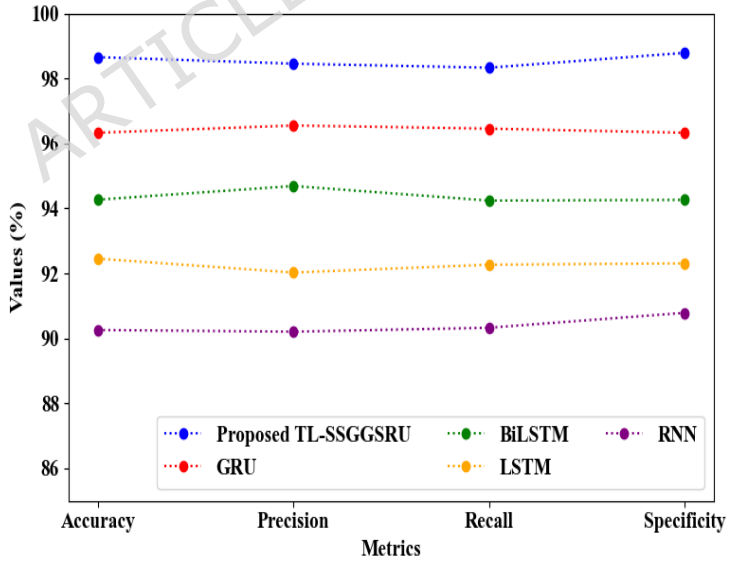

Figure 5: Performance validation for attack classification

Table 7: Comparative analysis of the proposed TL-SSGGSRU

Methods	Accuracy (%)	Precision (%)	Recall (%)	Specificity (%)	FPR	Brier score	Training time (ms)
Proposed TL-SSGGSRU	98.6524	98.3265	98.7845	98.5432	0.1254	0.0202	58343
GRU	96.3256	96.5487	96.4518	96.3259	1.5694	0.0634	70234
Bi LSTM	94.2656	94.8723	94.2592	94.5913	3.6813	0.0823	82311
LSTM	92.4510	92.2658	92.3062	92.4597	5.1245	0.0945	94368
RNN	90.2560	90.4325	90.7845	90.5322	7.6923	1.4114	114922

The proposed TL-SSGGSRU's performance is appraised with prevailing GRU, Bidirectional-LSTM (Bi-LSTM), LSTM, and RNN in Figure 5 and Table 7. Regarding accuracy, precision, recall, specificity, False Positive Rate (FPR), and training time, the TL-SSGGSRU attained 98.6523%, 98.4512%, 98.3265%, 98.7845%, 0.1254, and 58343ms, while the prevailing GRU obtained 96.3256%, 96.5487%, 96.4518%, 96.3259%, 1.5694, and 70234ms. The Brier score

is an evaluation metric used to validate the accuracy of probabilistic predictions. Here, the proposed TL-SSGGSRU obtained a Brier score of 0.002, whereas the traditional GRU and RNN attained a Brier score of 0.006 and 1.423, respectively. Thus, the proposed work had better performance due to the presence of SA. Swim activation improves the learning efficiency of the neurons by including smooth and non-saturating gradients. Moreover, the shake-shake regularization mitigates overfitting by regularizing the weight value effectively. In addition, the proposed work uses transfer learning, which aids in enhancing the generalizability of the model. However, the conventional classifiers had poor classifier performance owing to the lack of suitable weight regularization and activation function. Hence, the proposed model's superiority in attack classification was evidenced.

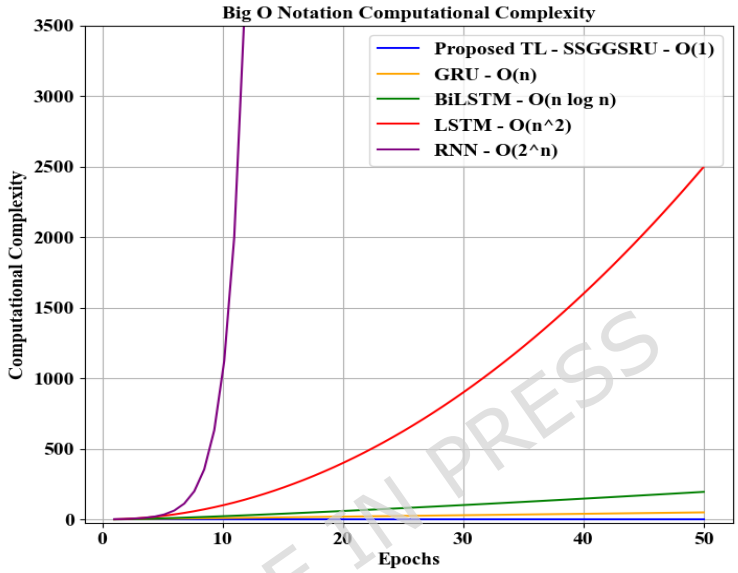

Figure 6: Computational complexity analysis

In Figure 6, the computational complexity of the proposed TL-SSGGSRU and existing classifiers is assessed. As the number of epochs increases, the proposed TL-SSGGSRU obtains stable computational complexity. By effectively regularizing the weight values using SSR, the proposed work had a minimum computational overhead. Improper weights cause slow convergence, thus requiring more training iterations. As a result, the model's complexity is significantly increased. Hence, the prevailing RNN had a sudden spike in complexity. Therefore, the proposed work was more effective in attack classification than existing classifiers.

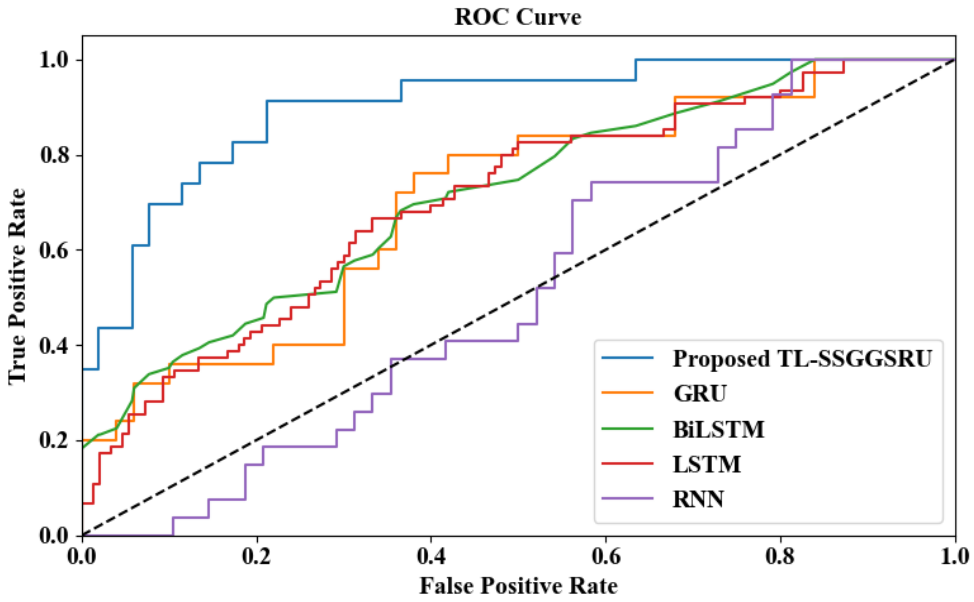

Figure 7: ROC curve analysis

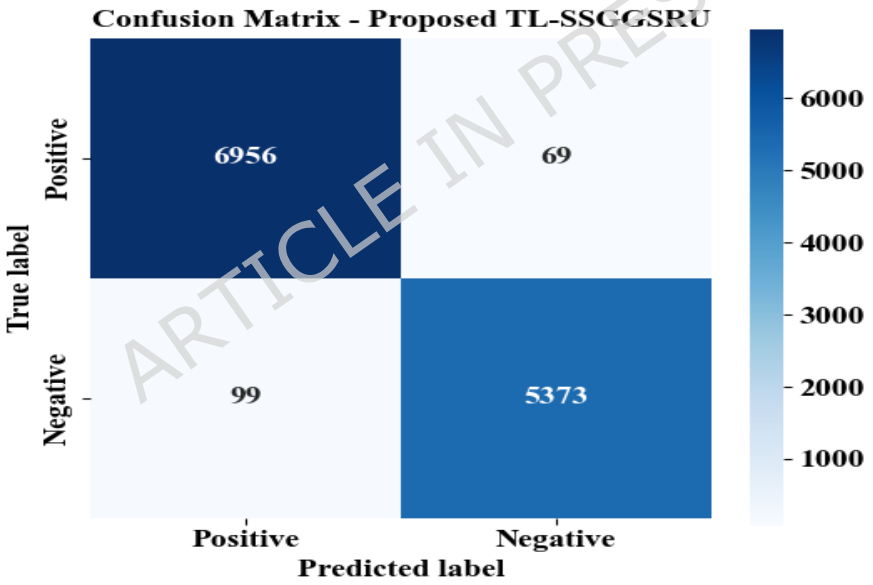

Figure 8: Confusion matrix of the proposed TL-SSGGSRU

The Receiver Operating Characteristic (ROC) curve analysis and confusion matrix analysis of the proposed TL-SSGGSRU are shown in Figures 7 and 8, respectively. ROC curve validates a binary classifier by visualizing the trade-off between true positive rate and false positive rate. The proposed TL-SSGGSRU obtained a high ROC curve, showing better classifier performance. However, the traditional RNN had ineffective performance due to the overfitting issues. Likewise, a confusion matrix is a performance metric used to validate a classification model by comparing actual vs predicted labels. Thus, the confusion matrix showed that the proposed TL-SSGGSRU accurately classified the true positives and false positives. Hence, the proposed method had more accurate results than existing algorithms.

The above-mentioned results were obtained with respect to the RDD. Further, the generalizability of the proposed TL-SSGGSRU in malware detection is proved by considering the additional datasets, including Elastic Malware Benchmark for Empowering Research – 2018, Version 2, Features (Ember-2018-V2-Features), Sophos-ReversingLabs 20 Million dataset (SOREL-20M), and Blue Hexagon Open Dataset for Malware Analysis (BODMAS). Ember-2018-V2-features is a benchmark malware dataset that involves static features extracted from Windows executables. Likewise, SOREL-20M is a large-scale malware detection dataset that includes labelled samples with respect to benign and malicious. Further, the BODMAS is an Android malware analysis dataset, encompassing bytecode, opcode sequences, and system information.

Table 8: Numerical investigation for attack classification regarding varying datasets

Techniques	**Accuracy (%)**	**Precision (%)**	**Recall (%)**
Ember-2018-V2-features			
Proposed TL-SSGGSRU	99.8089	99.8142	99.8074
GRU	96.3422	97.8934	96.9843
BiLSTM	94.3232	95.8394	93.8953
LSTM	92.9857	92.5738	92.5933
RNN	90.6754	89.7853	89.7864
SOREL-20M			
Proposed TL-SSGGSRU	98.9753	98.9678	98.9754
GRU	96.7334	96.6668	96.4677
BiLSTM	93.5783	94.7876	92.6776
LSTM	92.8954	92.5546	90.6765
RNN	89.7343	90.9768	87.8857

BODMAS			
Proposed TL-SSGGSRU	98.9585	98.9675	98.9764
GRU	96.6477	96.5648	96.5657
BiLSTM	95.3424	94.6773	94.5376
LSTM	92.6758	92.4767	92.7687
RNN	90.5776	89.5643	90.6743

In Table 8, the performance of the proposed TL-SSGGSRU and existing classifiers, like GRU, BiLSTM, LSTM, and RNN, is validated regarding varying datasets, including Ember-2018-V2-features, SOREL-20M, and BODMAS. The proposed TL-SSGGSRU obtained impressive outcomes in different datasets due to the inclusion of the Swim activation function and SSR-based weight regularization. Here, the proposed TL-SSGGSRU obtained accuracy, precision, and recall of 99.8089%, 99.8142% and 99.8074%, respectively, with respect to Ember-2018-V2-features. Similarly, the existing GRU had 96.3422% accuracy, 97.8934% precision, and 96.9843% recall. For the remaining datasets, the proposed work achieved a higher performance than existing classifiers. Hence, the generalizability of the proposed methodology was shown.

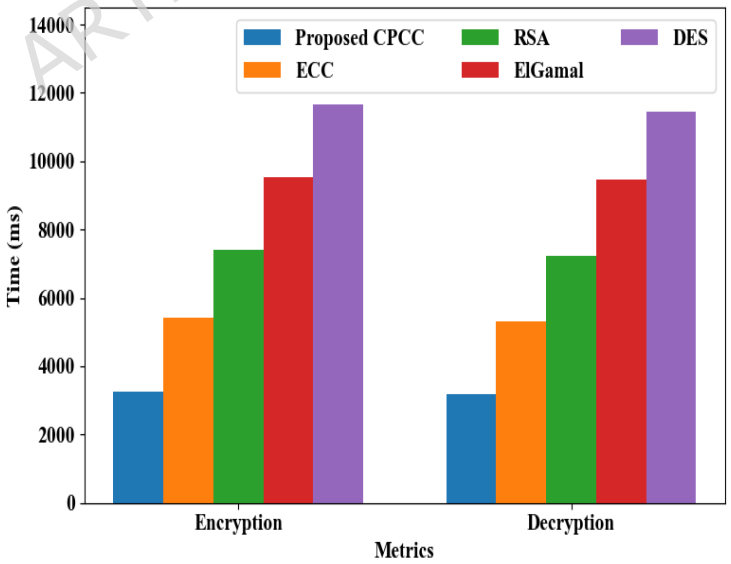

Figure 9: Performance analysis for data protection

Table 9: Numerical investigation of the proposed CPCC

Algorithms	Encryption Time (ms)	Decryption Time (ms)	Security Level (%)
Proposed CPCC	3245	3189	98.4578
ECC	5421	5326	95.3256
RSA	7415	7232	92.4578
ElGamal	9532	9451	89.6512
DES	11658	11451	86.6523

In Figure 9 and Table 9, the proposed CPCC's performance is evaluated with prevailing ECC, Rivest-Shamir-Adleman (RSA), ElGamal, and Data Encryption Standard (DES). Regarding encryption time and security level, the CPCC obtained 3245ms and 98.4578%; but, the prevailing techniques acquired 8506ms and 91.0217%. Likewise, the proposed CPCC obtained a decryption time of 3189ms, whereas the existing DES consumed 11451ms to decrypt the data. CPC reduces time complexity by computing curve parameters that neglect inefficient negative-point operations. Also, CPCC increases the randomness, thereby ensuring more resistance to attacks. Overall, the existing approaches had considerable time complexity and a poor security level due to the negative points in the traditional curves. Hence, the proposed work's high security level was proved.

Table 10: Experimental validation for node clustering

No of nodes / Methods	100	200	300	400	500
Energy Consumption (mj)					
Proposed PBCK-Means	1245	3454	5281	7122	9056
K-Means	3256	5062	7054	9563	11549
BIRCH	5084	7541	9233	11021	13625
CLARA	7124	9254	11054	13954	15748
FCM	9451	11248	13568	15847	17052
Latency (ms)					

Proposed PBCK-Means	2345	4519	6327	8145	10325
K-Means	4518	6322	8457	10658	12451
BIRCH	6084	8652	10784	12652	14693
CLARA	8125	10248	12329	14853	16284
FCM	10658	12953	14588	16872	18337

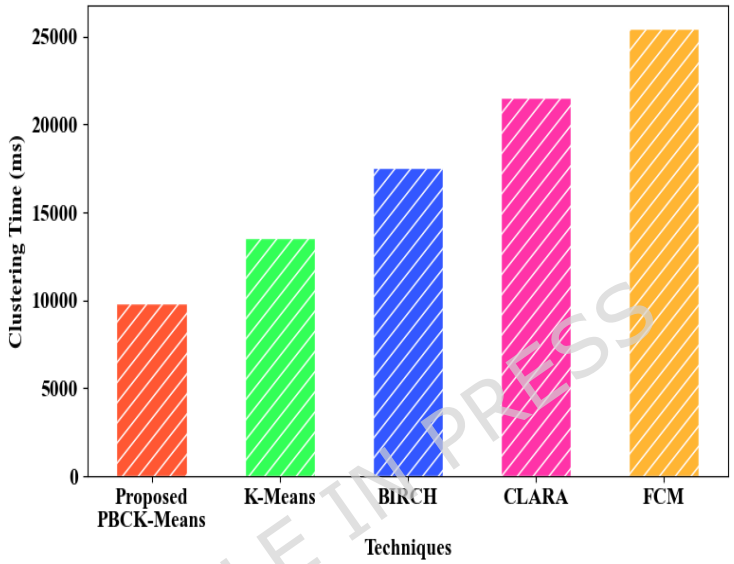

Figure 10: Clustering analysis

Table 10 and Figure 10 illustrate the empirical analysis of the proposed PBCK-Means and traditional techniques like K-Means, Balanced Iterative Reducing and Clustering using Hierarchies (BIRCH), Clustering of LARge Applications (CLARA), along with Fuzzy C-Means (FCM). The PBCK-Means achieved energy consumption and latency of 1245 milli joules (mj) and 2345ms for 100 nodes, respectively. Likewise, the proposed PBCK-Means took 9845ms to perform clustering, whereas the traditional K-Means consumed a clustering time of 13547ms. But, the conventional techniques had maximum energy consumption owing to the uneven-density data. In the proposed model, BCD improves clustering efficiency by measuring distribution overlap, thus ensuring well-separated clusters. Hence, the presence of PWI ensured better EE in the proposed work.

4.3 Ablation study

An ablation study assesses the impact of individual model components regarding model performance.

Table 11: Ablation study for attack classification

Techniques	Accuracy (%)	Precision (%)
Proposed TL-SSGGSRU	98.6523	98.4512
GRU + TL	98.1332	98.1637
GRU + Swim	98.1022	98.1133
GRU + SSR	97.8392	97.9932
GRU	96.3256	95.2045

Table 11 shows the ablation study of the proposed TL-SSGGSRU regarding attack classification. In the proposed work, TL is integrated with the attack classification module to adapt to the unknown threats, increasing the model's generalizability. Likewise, the swim activation effectively increases the neuron's learning efficiency by introducing non-linearity, mitigating the over-fitting issues. The GRU + Swim algorithm obtained accuracy and precision of 98.1022% and 98.1133%, correspondingly. Further, the proposed work employs the SSR to regularize the suitable weight values. Here, the GRU + SSR algorithm had 97.8392% accuracy and 97.9932% precision. But, the traditional GRU achieved a minimum accuracy and precision of 96.3256% and 95.2045%, correspondingly. Collectively, the proposed TL-SSGGSRU attained accuracy and precision of 98.6523% and 98.4512%, respectively. Hence, the importance of novel components, such as SSR regularization, Swim activation, and TL in attack classification, was proved.

Table 12: Empirical validation for node clustering

Energy consumption (mj)					
Techniques / Number of nodes	**100**	**200**	**300**	**400**	**500**
Proposed PBCK-Means	1245	3454	5281	7122	9056
K-Means + PWI	2383	3789	4985	6344	8953
K-Means + BCD	4783	6784	8974	9973	10493

K-Means	9451	11248	13568	15847	17052
Latency (ms)					
Proposed PBCK-Means	2345	4519	6327	8145	10325
K-Means + PWI	3167	4679	7193	8293	10983
K-Means + BCD	4222	4902	7534	8573	11083
K-Means	4518	6322	8457	10658	12451

In Table 12, the ablation study is conducted to prove the specific contributions of the proposed PBCK-Means algorithm. The utilization of PWI aids in estimating the data density to set initial centroids in high-probability regions, improving convergence. Likewise, BCD computes cluster separability to refine centroid updates, thus improving overall clustering efficiency. The proposed PBCK-Means achieved energy consumption and latency of 1245mj and 2345ms, respectively, regarding 100 nodes. However, the absence of PWI and BCD degrades the clustering outcomes. Here, the existing K-Means attained energy consumption and latency of 9451mj and 4518ms, respectively, according to 100 nodes. Hence, the significant contributions of the proposed components, such as PWI and BCD, were evidenced.

Table 13: Experimental validation for proposed work

Scenario	**End-to-End latency per GB (ms)**	**End-to-End throughput per GB (Mbps)**	**Compute energy per GB (mj)**	**Communication energy per GB (mj)**
Normal download	1248	4895	1242	1278
Attack-verify-restore	2200	3989	1342	1789

Table 13 exhibits the performance assessment of the proposed work regarding end-to-end latency, end-to-end throughput, compute energy, and communication energy with respect to

two scenarios: normal download and attack-verify-restore. During normal conditions, the proposed work achieved end-to-end latency per GB, end-to-end throughput per GB, compute energy per GB, and communication energy per GB of 1248ms, 4895Mbps, 1242mj, and 1278mj, respectively. In the same manner, the proposed framework obtained end-to-end latency per GB, end-to-end throughput per GB, compute energy per GB, and communication energy per GB of 2200ms, 3989Mbps, 1342mj, and 1789mj, correspondingly. In the proposed work, BDI verification was done using a robust MPH that obtains high speed and robustness. In the proposed work, a novel CPCC was utilized to encrypt the sensitive information during transmission, thereby mitigating the impact of tampering. As a result, overall energy consumption was minimized. Furthermore, the proposed PBCK-Means significantly grouped the nodes, thus minimizing the energy consumption and end-to-end latency. Therefore, the proposed work had significant performance in both the normal and verification scenarios.

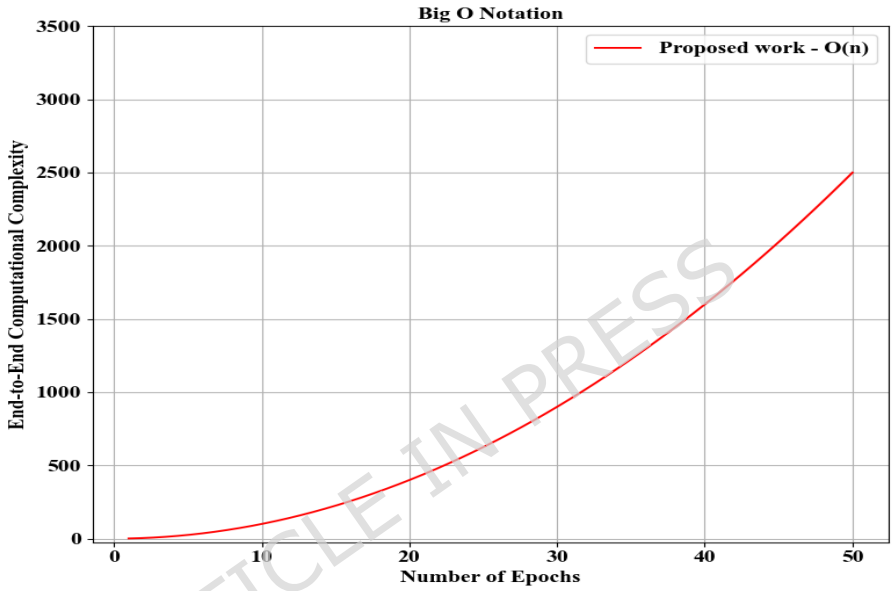

Figure 11: End-to-End Computational complexity analysis

Figure 11 illustrates the end-to-end computational complexity of the proposed framework. The presence of SSR and the swim activation function reduces the overfitting issues and vanishing gradient issues. As the number of epochs increased, the proposed work had a gradual increase in computational complexity. As a result, computational overhead was minimized. Moreover, the proposed PBCK-Means effectively grouped the network nodes, thus reducing the complexity during transmission. In the research methodology, the proposed CPCC was utilized to encrypt the sensitive information, thereby minimizing the risk of tampering and re-transmission. Hence, the proposed work had minimum end-to-end computational complexity.

Table 14: Empirical analysis regarding varying attack scenarios

Membership inference	
Techniques	Accuracy (%)
Proposed TL-SSGGSRU	98.9738
GRU	96.8922
BiLSTM	94.2322
LSTM	92.8933
RNN	90.7283

Rollback	
Proposed TL-SSGGSRU	98.9066
GRU	96.6752
BiLSTM	94.2783
LSTM	92.6472
RNN	89.6753
Ransomware evasion	
Proposed TL-SSGGSRU	98.9665
GRU	95.8322
BiLSTM	94.2392
LSTM	92.8933
RNN	90.8922

Table 14 showcases the empirical analysis of the proposed TL-SSGGSRU and existing classifiers regarding various attack scenarios: rollback, membership inference, and ransomware evasion. Regarding rollback, the proposed TL-SSGGSRU obtained an accuracy of 98.9066%, whereas the traditional RNN attained an accuracy of 89.6753%. Overall, the proposed methodology achieved better performance under varying attack scenarios.

4.3 Comparative evaluation of the research methodology

Here, the comparative assessment is done by comparing the performance of the proposed work with several recent extant studies with respect to network intrusion detection.

Table 15: Comparative validation of the proposed work

Authors' name	Algorithms	Accuracy (%)	Precision (%)
Proposed work	TL-SSGGSRU	99.80	99.81
Urooj et al. [21]	wGAN and ensemble DL	97.65	97.02
Al-Hawawreh et al. [22]	DNN with batch normalization	98.13	98.20
Routray et al. [23]	Ensemble machine learning	98.26	60.00
Zahoora et al. [24]	CSPES	93.00	-
Gurukala & Verma [25]	RF + SVM	93.82	93.57
Nandanwar & Katarya [26]	GAO-XGBoost	98.00	-
(Nandanwar & Katarya [27]	Optimized extreme gradient boosting algorithm	98.12	-

(Nandanwar & Katarya [28]	Interpretable Cyber-Sentinet	97.46	97.7
(Nandanwar & Katarya [29]	CNN-BiLSTM with transfer learning-BiLSTM	99.52	-
(Nandanwar & Katarya [30]	Adaptive CNN-GRU	99.75	99.75

The performance of the proposed work and related studies is compared in Table 15. The proposed TL-SSGGSRU had 98.65% accuracy and 98.45% precision in attack classification. The inclusion of TL aided in improving the adaptability and efficiency of the proposed work in recognizing the RA. Similarly, the conventional works utilized a weighted Generative Adversarial Network (wGAN) with ensemble DL [21], Deep Neural Network (DNN) [22], and Cost-Sensitive Pareto Ensemble Strategy (CSPES) [24] to perform RA prediction. In addition, the existing studies utilized various algorithms, such as the Genetic Algorithm-Optimized XGBoost (GAO-XGBoost) model, the optimized extreme gradient boosting approach, interpretable cyber-sentinet, CNN-BiLSTM with transfer learning, and adaptive CNN-GRU, to perform network intrusion detection. The existing interpretable cyber-Sentinet [28] obtained accuracy and precision of 97.46% and 97.7%, respectively. Here, the traditional RF and Support Vector Machine (SVM) [25] achieved an accuracy of 93.82% and precision of 93.57% in RA detection. Due to improper activation functions, the existing works had poor attack detection performance. Thus, the proposed approach had higher significance in RA classification.

5. CONCLUSION

Here, an improved AI-assisted BDI verification scheme for CR in GSI is proposed. The proposed MPH was utilized to authenticate the integrity of the backup data before restoration, ensuring high resilience. A novel TL-SSGGSRU was generalized well enough to classify the data into attack or normal with high adaptability. Furthermore, processes like data protection, node clustering, and correlation heatmap generation aided in improving the security level and EE. The inclusion of transfer learning aided in classifying the unknown threats, thereby improving the model's superiority. Therefore, as per the outcome, the proposed work obtained an accuracy and security level of 98.65% and 98.45%, respectively, showing high security and efficacy. Overall, the proposed methodology significantly contributed to enhancing the backup integrity and network security in the sustainable smart infrastructure.

Limitations – In real-time, the proposed work faces several challenges related to adaptability and computational efficiency. The proposed methodology may struggle to handle the dynamic behaviour of the large-scale network during attack classification. Also, the proposed approach primarily focused on backup integrity and ransomware classification rather than predicting combined or evolving attack vectors. To maintain better accuracy, the proposed attack

classification model requires frequent retraining. In a real-world large-scale environment, distributed backup storage may introduce latency and energy overhead.

Future scope: Thus, this work will focus on classifying the various cyberthreats, including denial-of-service, password hijacking, and zero-day attacks, using advanced ensemble approaches to further increase cyber-resilience in the future. To mitigate adaptability issues, future work will integrate self-evolving learning strategies. Also, lightweight model updation and FL will be established to reduce frequent retraining dependency in dynamic smart infrastructure networks. Furthermore, edge-accelerated integrity checks will be introduced to minimize the latency and energy overhead issues.

Data Availability

The datasets generated and/or analysed during the current study are available in the **Dataset:** https://www.kaggle.com/datasets/amdj3dax/ransomware-detection-data-set

Author Contributions

Formal Analysis-Basant Kumar, Shashi Kant Gupta and Ozlem Kilickaya; Resources, Design and coding : Rashmi Dwivedi; Wrting-Original Draft- Diaa Salama AbdElminaam; Writing-Deema Mohammed Alsekait; Review and Editing- Basant Kumar

Funding Statement

This research was funded by Princess Nourah bint Abdulrahman University Researchers Supporting Project number (PNURSP2025R435), Princess Nourah bint Abdulrahman University, Riyadh, Saudi Arabia

ACKNOWLEDGMENT

The authors would like to acknowledge the support of Princess Nourah bint Abdulrahman University Researchers Supporting Project number (PNURSP2025R435), Princess Nourah bint Abdulrahman University, Riyadh, Saudi Arabia.

REFERENCES:

1. Konstantinou, C. Toward a Secure and Resilient All-Renewable Energy Grid for Smart Cities. *IEEE Consum. Electron. Mag.* **11**(1), 1–7, DOI: https://doi.org/10.1109/MCE.2021.3055492 (2021).

2. Saheed, Y. K. & Arowolo, M. O. Efficient Cyber Attack Detection on the Internet of Medical Things-Smart Environment Based on Deep Recurrent Neural Network and Machine Learning Algorithms. *IEEE Access*, **9**, 161546–161554, DOI: https://doi.org/10.1109/ACCESS.2021.3128837 (2021).

3. Rana, M. U., Shah, M. A., Alnaeem, M. A. & Maple, C. Ransomware Attacks in Cyber-Physical Systems: Countermeasure of Attack Vectors Through Automated Web Defenses. *IEEE Access*, **13**, 1–18, DOI: https://doi.org/10.1109/ACCESS.2024.3477631 (2024).

4. Malik, M. I., Ibrahim, A., Hannay, P. & Sikos, L. F. Developing Resilient Cyber-Physical

Systems: A Review of State-of-the-Art Malware Detection Approaches, Gaps, and Future Directions. *Computers*, **12**(4), 1–26, DOI: https://doi.org/10.3390/computers12040079 (2023).

5. Bhandari, G. P., Lyth, A., Shalaginov, A. & Gronli, T. M. Artificial Intelligence Enabled Middleware for Distributed Cyberattacks Detection in IoT-based Smart Environments. *Proceedings - 2022 IEEE International Conference on Big Data, Big Data 2022*, 3023–3032, DOI: https://doi.org/10.1109/BigData55660.2022.10020531 (2022).

6. Madhavan, K., Yazdinejad, A., Zarrinkalam, F. & Dehghantanha, A. Security Vulnerabilities: A Metric-Driven Security Analysis of Gaps in Current AI Standards. *arXiv preprint arXiv:2502.08610*, DOI: https://doi.org/10.48550/arXiv.2502.08610 (2025).

7. Dehghantanha, A., Yazdinejad, A. & Parizi, R. M. Autonomous cybersecurity: Evolving challenges, emerging opportunities, and future research trajectories. In *Proceedings of the Workshop on Autonomous Cybersecurity*, 1-10, DOI: https://dl.acm.org/doi/abs/10.1145/3689933.3690832 (2023).

8. Ahmed, Y., Beyioku, K. & Yousefi, M. Securing smart cities through machine learning: A honeypot-driven approach to attack detection in Internet of Things ecosystems. *IET Smart Cities*. **6**(3), 180–198, DOI: https://doi.org/10.1049/smc2.12084 (2024).

9. Khattak, Z. H. Cybersecurity vulnerability and resilience of cooperative driving automation for energy efficiency and flow stability in smart cities. *Sustain. Cities Soc.* **106**, 1–19, DOI: https://doi.org/10.1016/j.scs.2024.105368 (2024).

10. Lyu, Q., Liu, S. & Shang, Z. Securing Urban Landscape: Cybersecurity Mechanisms for Resilient Smart Cities. *IEEE Access*, **13**, 10966–10977. DOI: https://doi.org/10.1109/ACCESS.2024.3522078 (2025).

11. Dhingra, B., Jain, V., Sharma, D. K., Gupta, K. D. & Kukreja, D. RLET: a lightweight model for ubiquitous multi-class intrusion detection in sustainable and secured smart environment. *Int. J. Inf. Secur.* **23**(1), 1–17, DOI: https://doi.org/10.1007/s10207-023-00739-2 (2024).

12. Usman, Y., Ihejirika, C. J., Offor, S. N., Robert, A. & Chataut, R. Green Cybersecurity: Leveraging AI, ML, and LLMs to Optimize Energy, Threat Detection, and Sustainability Frameworks. *IEEE Access*, 1–35, DOI: https://doi.org/10.1109/ACCESS.2025.3602451 (2025).

13. Ali, S. M., Razzaque, A., Yousaf, M. & Ali, S. S. A Novel AI-Based Integrated Cybersecurity Risk Assessment Framework and Resilience of National Critical Infrastructure. *IEEE Access*. **13**, 12427–12446, DOI: https://doi.org/10.1109/ACCESS.2024.3524884 (2025).

14. Alazab, M., Khurma, R. A., Garcia-Arenas, M., Jatana, V., Baydoun, A. & Damaševičius, R. (2024). Enhanced threat intelligence framework for advanced cybersecurity resilience. Egypt. Inform. J. **27**, 1–26, DOI: https://doi.org/10.1016/j.eij.2024.100521 (2024).

15. Islam, S., Basheer, N., Papastergiou, S., Ciampi, M. & Silvestri, S. Intelligent dynamic cybersecurity risk management framework with explainability and interpretability of AI models for enhancing security and resilience of digital infrastructure. *J. Reliab.*

*Intell. Environ.***11**(3), 1–25, DOI: https://doi.org/10.1007/s40860-025-00253-3 (2025).

16. Zhang, X., Wang, J. & Zhu, S. Dual Generative Adversarial Networks Based Unknown Encryption Ransomware Attack Detection. *IEEE Access*, **10**, 900–913, DOI: https://doi.org/10.1109/ACCESS.2021.3128024 (2022).

17. Singh, A., Mushtaq, Z., Abosaq, H. A., Mursal, S. N. F., Irfan, M. & Nowakowski, G. Enhancing Ransomware Attack Detection Using Transfer Learning and Deep Learning Ensemble Models on Cloud-Encrypted Data. *Electronics (Switzerland)*, *12*(18), 1–31. https://doi.org/10.3390/electronics12183899 (2023).

18. Nandanwar, H. & Katarya, R. A secure and privacy-preserving ids for iot networks using hybrid blockchain and federated learning. In *International Conference on Next-Generation Communication and Computing*, 207-219, DOI: https://link.springer.com/chapter/10.1007/978-981-96-3725-6_18 (2025c).

19. Yazdinejad, A., Mohammadabadi, Z. D., Dehghantanha, A., & Srivastava, G. An Explainable and Privacy-Preserving Federated Learning Model for Threat Detection in Cyber-Physical-Social Systems. *IEEE Trans. Comput. Soc. Syst.* 1 – 13, DOI: https://doi.org/10.1109/TCSS.2025.3606798 (2025).

20. Yazdinejad, A., Dehghantanha, A., Karimipour, H., Srivastava, G. & Parizi, R. M. A robust privacy-preserving federated learning model against model poisoning attacks. *IEEE Trans. Inf. Forensics Secur.* **19,** 6693-6708, DOI: https://doi.org/10.1109/TIFS.2024.3420126, (2024).

21. Urooj, U., Al-Rimy, B. A. S., Zainal, A. B., Saeed, F., Abdelmaboud, A. & Nagmeldin, W. Addressing Behavioral Drift in Ransomware Early Detection Through Weighted Generative Adversarial Networks. *IEEE Access*, **12,** 3910–3925, DOI: https://doi.org/10.1109/ACCESS.2023.3348451 (2024).

22. Al-Hawawreh, M., Sitnikova, E. & Aboutorab, N. Asynchronous peer-to-peer federated capability-based targeted ransomware detection model for industrial IoT. *IEEE Access*. **9**, 148738–148755, DOI: https://doi.org/10.1109/ACCESS.2021.3124634 (2021).

23. Routray, S., Prusti, D. & Rath, S. K. Ransomware Attack Detection by Applying Machine Learning Techniques. *Lect. Notes Electr. Eng.* **1,** 1–12, DOI: https://doi.org/10.1007/978-981-99-0085-5_62 (2023).

24. Zahoora, U., Khan, A., Rajarajan, M., Khan, S. H., Asam, M. & Jamal, T. Ransomware detection using deep learning based unsupervised feature extraction and a cost sensitive Pareto Ensemble classifier. *Sci. Rep.* **12**(1), 1–15, DOI: https://doi.org/10.1038/s41598-022-19443-7 (2022).

25. Gurukala, N. K. Y. & Verma, D. K. Feature Selection Using Particle Swarm Optimization and Ensemble-Based Machine Learning Models for Ransomware Detection. *SN Comput. Sci.* **5**(8), 1–21. DOI: https://doi.org/10.1007/s42979-024-03454-4 (2024).

26. Nandanwar, H. & Katarya, R. Optimized intrusion detection and secure data management in IoT networks using GAO-Xgboost and ECC-integrated blockchain framework. *Knowl. Inf. Syst.* 1-56, DOI: https://link.springer.com/article/10.1007/s10115-025-02513-3 (2025).

27. Nandanwar, H. & Katarya, R. A hybrid blockchain-based framework for securing intrusion detection systems in internet of things. Clust. Comput. **28**(7), 471, DOI: https://link.springer.com/article/10.1007/s10586-025-05135-0 (2025a).

28. Nandanwar, H. & Katarya, R. Securing Industry 5.0: An explainable deep learning model for intrusion detection in cyber-physical systems. *Comput. Electr. Eng.* **123**, 110161, DOI: https://doi.org/10.1016/j.compeleceng.2025.110161 (2025b).

29. Nandanwar, H. & Katarya, R. TL-BILSTM IoT: transfer learning model for prediction of intrusion detection system in IoT environment. *Int. J. Inf. Secur.* **23**(2), 1251-1277, DOI: https://link.springer.com/article/10.1007/s10207-023-00787-8 (2023).

30. Nandanwar, H. & Katarya, R. Deep learning enabled intrusion detection system for Industrial IOT environment. *Expert Syst. Appl.* **249**, 123808, DOI: https://www.sciencedirect.com/science/article/abs/pii/S0957417424006742 (2024).