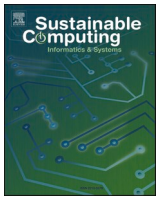
Sustainable Computing: Informatics and Systems 46 (2025) 101127

Contents lists available at ScienceDirect

Sustainable Computing: Informatics and Systems

journal homepage: www.elsevier.com/locate/suscom



A predominant intrusion detection system in IIoT using ELCG-DSA AND LWS-BiOLSTM with blockchain

Basava Ramanjaneyulu Gudivaka [a,*], Rajya Lakshmi Gudivaka [b], Raj Kumar Gudivaka [c], Dinesh Kumar Reddy Basani [d], Sri Harsha Grandhi [e], Sundarapandian Murugesan [f], M.M. Kamruzzaman [g]

[a] Raas Infotek, Delaware, USA
[b] Wipro, Hyderabad, India
[c] Infosys, Charlotte, USA
[d] CGI, British Columbia, Canada
[e] Intel, Folsom, CA, USA
[f] eRay Technologies LLC, NJ, USA
[g] Department of Computer Science, College of Computer and Information Sciences, Jouf University, Sakakah, Saudi Arabia

ARTICLE INFO

Keywords:
Blockchain
Intrusion Detection System (IDS)
Deep Learning (DL)
IIoT, Hash Code Generation
Industry Control System
Digital Signature Creation
Power-aware Applications

ABSTRACT

The growing connectivity of Industrial Internet of Things (IIoT) systems has increased cyber threats, necessitating early detection of intrusions. However, existing systems often lack focus on intermediate and continuous multifactor authorization between IIoT and Industrial Control Systems (ICS). To overcome this, an efficient IDS for IIoT using an Exponential Linear Congruential Generator - Digital Signature Algorithm (ELCG-DSA) and Log Wave Sigmoid-Bidirectional Once Long Short-Term Memory (LWS-BiOLSTM) is proposed. Initially, the industry and vehicle details are registered in the blockchain network, and the Polychoric Entropy Correlation-Tiger Hashing Algorithm (PEC-Tiger) generates hash codes through smart contract creation. From the generated hash codes, a partial digital signature is created by using the ELCG-DSA technique. After login, the registered details are processed for enhancing security using Montgomery Modulo Curve Cryptography (MMCC). Then, the details are verified by using PEC-Tiger, and if the hash code matches, the key generation centre is notified for the creation of a fully digital signature. After verification, the Luus–Jaakola Sequence-based Pelican Optimization Algorithm (LJS-POA) is applied for load balancing. Next, the data security is verified in the IDS training set, in which the features are extracted from preprocessed data. Then, the Synthetic Minority Oversampling Technique (SMOTE) is utilized for data balancing, and LWS-BiOLSTM is implemented to classify attacks. Furthermore, the attacked data is blocked, and non-attacked data is stored in the ICS through digital signature verification. Thus, the experimental results of the proposed framework outperform the other conventional techniques by achieving 98.78 % accuracy and 98.71 % security level.

1. Introduction

The IIoT states the extension and application of the IoT in industrial sectors [22]. IIoT involves the integration of sensors, software, and other technologies with industrial equipment and processes [28] to collect, analyze, and exchange data [17]. High-dimensional imaging data, such as MRI scans, may be preprocessed using the framework by employing convolutional layers to extract features. It can improve real-time threat identification during MRI data transfers in telemedicine and anomaly detection in healthcare. Use image libraries like OpenCV or ITK and train on hybrid datasets to implement this. Additionally, IIoT improves decision-making and operational efficiency through real-time data collection, analysis, and automation [2]. But, IIoT introduces security challenges due to the proliferation of connected devices [9]. The challenges include cyber-attack vulnerabilities, data breaches, and unauthorized access [25]. Thus, it risks sensitive industrial systems and

* Corresponding author.
E-mail addresses: basavagudivaka@ieee.org (B.R. Gudivaka), rlakshmigudivaka@ieee.org (R.L. Gudivaka), rajkumargudivaka@ieee.org (R.K. Gudivaka), dineshkumarreddybasan@ieee.org (D.K.R. Basani), sriharshagrandhi@ieee.org (S.H. Grandhi), sundarapandianmurugesan@ieee.org (S. Murugesan), m.m.kamruzzaman@ieee.org, mmkamruzzaman@ju.edu.sa (M.M. Kamruzzaman).

https://doi.org/10.1016/j.suscom.2025.101127
Received 28 November 2024; Received in revised form 21 March 2025; Accepted 10 April 2025
Available online 19 April 2025
2210-5379/© 2025 Elsevier Inc. All rights are reserved, including those for text and data mining, AI training, and similar technologies.

critical infrastructure [10]. Consequently, there is a pressing need to detect intrusions [13] to monitor network traffic security gaps, ensuring the integrity and security of the IIoT environment [20]. IDS plays a significant role in preserving the IIoT networks and assets' integrity and security [21].

Moreover, blockchain-based IDS leverages blockchain's decentralized and immutable nature to enhance security [24]. They offer increased transparency, trust, and resilience against cyber threats in industrial environments [8]. Furthermore, blockchain-based IDS provides security measures, but the incorporation of Machine Learning (ML) and Deep Learning (DL) methodologies enhances defence mechanisms in IIoT environments [16]. ML models like Random Forest (RF) offer better accuracy in detecting anomalies but may require significant computational resources [12]. Subsequently, DL models like Convolutional Neural Networks (CNNs) excel in identifying complex patterns and anomalies but necessitate huge amounts of labelled data for training [6]. So, to overcome these limitations, an efficient framework is proposed for IDS in an IIoT environment using ELCG-DSA and LWS-BiOLSTM. Network security depends on intrusion detection systems (IDS), but existing solutions have issues with scalability, sluggish reaction times to new threats, unbalanced datasets, and excessive computing overhead. Current methods are ineffective at addressing encrypted or polymorphic assaults, have slow reaction times, are biased towards majority classes, and are not very flexible regarding new and changing threats.

1.1. Problem statement

Several prevailing works had some limitations, which are illustrated as follows,

- Existing works do not prioritize intermediate and continuous multifactor authorization between IIoT and ICS to prevent attacks.
- Most of the conventional approaches exhibit improper results due to the utilization of an unbalanced dataset.
- Many traditional techniques have low response times during transmitting IIoT data to the server through sensing owing to traffic congestion.
- Data security limitations arise, affecting sensitive information's reliability and confidentiality in some prevailing techniques.

The major objectives of the proposed framework are depicted further,

- The proposed framework applied ELCG-DSA to mitigate attacks for intermediate and continuous multifactor authorization between IIoT and ICS.
- The SMOTE algorithm is implemented to effectively address the class imbalance issue by generating synthetic samples for minority classes, enhancing performance.
- The proposed LJS-POA technique is established to address the limitation of low response.
- The proposed framework utilized MMCC for enhancing security and making it well-suited for modern digital security.

The outline of this paper is illustrated as follows: Section 2 indicates the literature survey, Section 3 depicts the proposed framework, Section 4 demonstrates the results and discussion, and Section 5 concludes the paper with future scope.

2. Literature survey

Rathee et al., [19] described a blockchain-centric IDS utilizing indirect trust intended for IIoT systems. The study used the Viterbi algorithm for the blockchain mechanism, and Industrial Internet of Things (IIoT) systems were employed to gauge the malicious activities' probability in the network. The Viterbi, as well as indirect methods, assessed system transparency. Results showed improved IIoT performance, with the applied learning technique reducing computational steps. However, a limitation was that the study did not address the potential scalability issues and real-time implementation challenges.

Devarjan et al. [5] Tackles intricate data gathering and threat detection difficulties by introducing a Recurrent Rule-based Feature Selection (RFS) for IIoT systems. By employing a hybrid rule-based algorithm to classify data and forecast assaults, the RFS model achieves low false positive rates, greater accuracy rates of 99.0 % and 98.9 %, and detection rates of 99.0 % and 99.9 %.

Wu et al. [33] explained a secure and scalable IIoT system using the convergence of blockchain and edge computing. The study examined cloud capabilities using the IIoT critical infrastructure method, followed by utilizing a distributed system of blockchain and edge computing paradigms to secure ledger transactions. The convergence paradigms enabled secure and scalable critical infrastructures. Results indicated higher effectiveness values for the IIoT blockchain. However, a limitation was the IoT devices' resource constraints, which stopped the adoption of blockchain in IIoT.

Shyam Mohan et al. [27] Network risks and attack stages may only be comprehended by conducting a network security situation assessment (NSSA). An intrusion alarm prediction system based on blockchain technology is created for cyber-security threat intelligence (CTI) and situational awareness. Alert segregation, a blockchain-based CTI model, and hybrid Soergel and Lorentzian characteristics are all used in the system. Intrusion alerts are predicted using the Deep Maxout Network (DMN), and blacklisting is utilized to mitigate cyberattacks. With better recall, accuracy, and F-measure, the updated DMN performs better.

Dat-Thinh et al. [4] Introduced a multistage IDS using the IoT. Initially, the internet gateways, together with IoT local gateways, were deployed by a collaborative IDS. Subsequently, the IoT network classified the type of each IoT device and identified the type of attacks targeting them. Finally, the internet gateways managed the two stages. Results demonstrated a higher accuracy rate, but the applied method faced limitations due to computational capacity constraints.

To tackle the problem of widespread cyber threats in smart factories, Kalyan Gattupalli and Poovendran Alagarsundaram et al. [26] present an advanced Faster Recurrent Convolutional Neural Network (Faster R-CNN) coupled with an edge computing-based malware identification model. The system can identify malware by sending many IIoT traffic data to edge servers. Accuracy, recall, and precision were 93.77 %, 95.87 %, 86.66 %, and 91.03 %, respectively, for the model.

Vargas et al. [32] Explained identifying security attacks in IIoT networks using a blockchain and ML model. Initially, IoT device networks integrated preceding solutions to develop an integral protection system. Subsequently, IIoT identified computational capabilities and threat identification and activated secure information transfer mechanisms. A machine learning approach was then employed to contain intruders in the IoT network. Results showed higher accuracy in calculating malicious traffic time. However, the performance of the applied method exhibited minimal fluctuation due to load and scalability issues.

Swapna [29] suggests a blockchain-based approach that uses Homomorphic Verifiable Tags (HVT) and chain code to ensure data integrity in multi-cloud storage systems. This approach uses cryptographic promises, system modelling, Data Owners (DOs), Cloud Service Providers (CSPs), and a Blockchain Network to guarantee data security and dependability. An experimental assessment of a standard computer configuration validates the efficiency and scalability of the technique. The method boosts trust through decentralized verification and establishes the foundation for further study into system resilience and cutting-edge security techniques.

Salim et al. [23] explained a blockchain-aided secure Digital Twin (DT) methodology for detecting botnets earlier in IIoT environments. Initially, the blockchain-enabled digital framework detected bot

creation early in a smart factory environment. Subsequently, DT was considered for a group of edge devices to collect data using a deep learning algorithm. Smart contracts authenticated data synchronized betwixt the DT and a Packet Auditor (PA) to detect corrupt device data transmission. Results showed that improved packet inspection yield was attained. However, the twin framework had limitations regarding precise data collection periods and storage capacity.

Dharma Teja [31] developed a secure and scalable HRM data management system integrating blockchain, AI, and Sparse Matrix Decomposition to enhance data security, scalability, and decision-making. This system ensures secure data storage, uses AI for predictive insights, and efficiently handles large datasets. Nevertheless, our research applies AI to improve HR data management by combining blockchain for secure storage, AI for predictive analytics, and Sparse Matrix Decomposition for handling large, incomplete datasets, enhancing real-time monitoring, scalability, and decision-making in HR systems.

Alosaimi, Almutairi [1] proffered an IDS based on BoT-IoT, focusing on quick and accurate attack prediction in IoT networks. A novel approach combining DL and three-level algorithms was utilized. The Bot-IoT dataset was employed for data collection, and the BoT-IoT method was extended to augment security in other IoT applications. Results showed higher performance of indicators. However, the high dependence on IoT and its rapid growth increased security risks, highlighting the crucial need for network security solutions.

Kadiyala, Kaur [11] developed a security-enhanced system for IoT data sharing that combines isogeny-based hybrid cryptography, anisotropic random walks (ARW), and decentralized cultural co-evolutionary optimization (DCCO) to improve data security. This approach optimizes encryption, strengthens data transmission security, and enhances threat detection in IoT systems. In the same way, our study employs an AI-powered system that combines isogeny-based cryptography for encryption, DCCO for dynamic optimization, and ARW for secure transmission to improve data security, scalability, and resilience. Like Kadiyala and Kaur's technique, this one guarantees safe data exchange while enhancing IIoT settings' resilience to cyberattacks, data integrity, and real-time monitoring.

Tharewal et al. [30] described an IDS for IIoT regarding deep reinforcement learning. Initially, the method integrated decision-making abilities to enhance the learning process. Subsequently, the near-end strategy optimization method was applied for IIoT intrusion detection. The DRL IDS utilized a feature selection method in terms of light GBM. Results indicated higher accuracy rates. However, the limitation noted here was the absence of a distributed architecture in IoT intrusion detection systems.

3. Proposed methodology for ELCG-DSA and LWS-BiOLSTM based IDS IN IIoT

This section demonstrates the proposed IDS in IIoT using ELCG-DSA and LWS-BiOLSTM. Fig. 1 illustrates the proposed model's block diagram. The LWS-BiOLSTM and ELCG-DSA are sophisticated methods for identifying abnormalities and large-scale coordinated assaults in IIoT systems. With their increased learning accuracy for sequential data streams, real-time distributed security monitoring, and threat detection capabilities, they successfully tackle latency, complexity, and scalability issues.

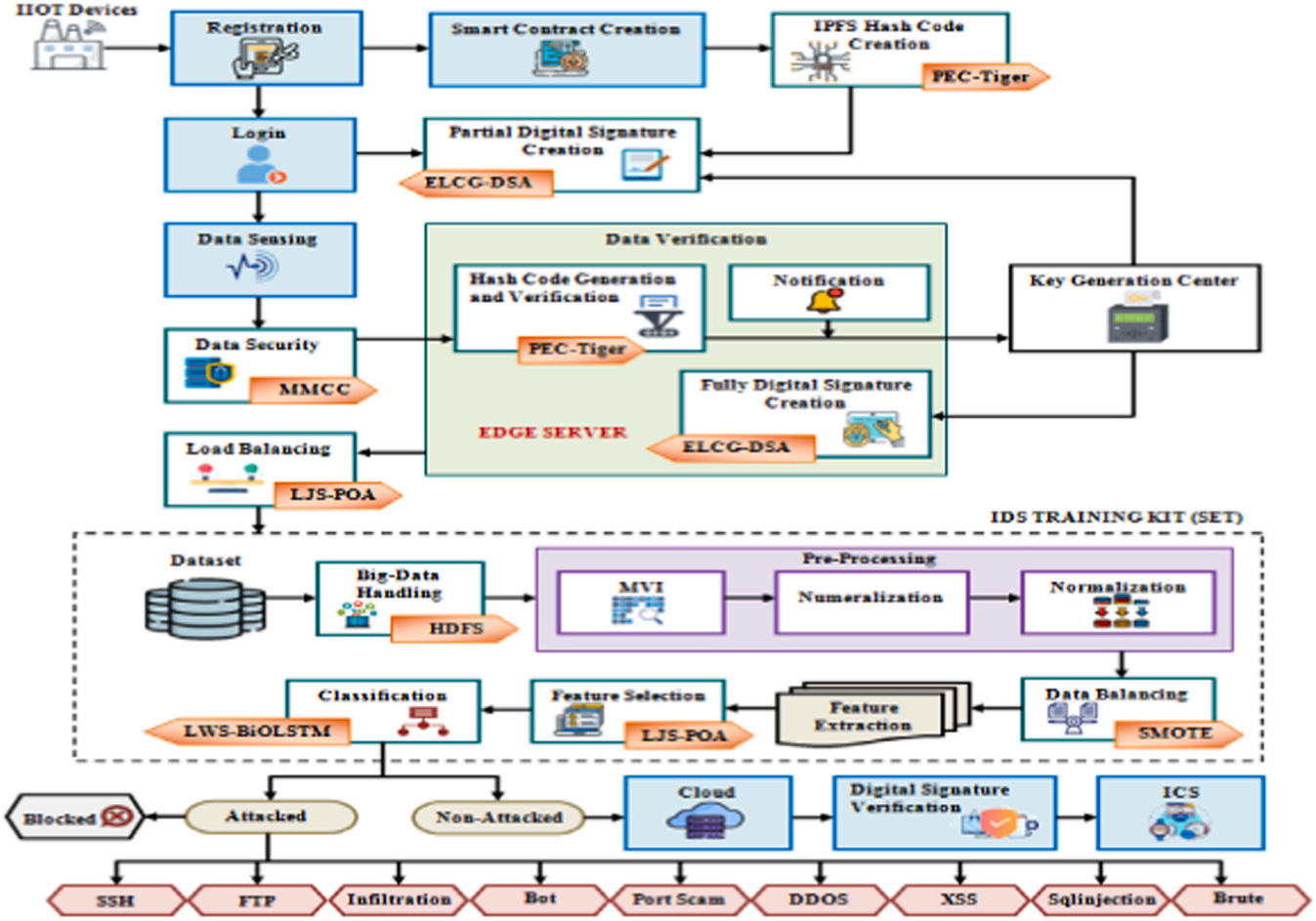


Fig. 1. Block diagram for the proposed framework.

3.1. Registration

Initially, industry and device details are collected from IIoT devices, and then, the details are registered in the blockchain network for secure transmission. Thus, the registered details are represented as$\eta\vartheta$.

$$\eta\vartheta_N = \{\eta\vartheta_1, \eta\vartheta_2, \eta\vartheta_3, \cdots, \eta\vartheta_n\}; N = 1 \rightarrow n \tag{1}$$

Where n is the number of registered details, a Global Unique Identifier (GUID) is assigned to each device at registration. Once the registration is successful, a smart contract is created between IIoT and ICS, and the created smart contract is represented as$\Re\beta$.

3.2. IPFS hash code creation

Based on$\Re\beta$, the Interplanetary File System (IPFS) hash codes are created using the proposed PEC-Tiger. The PEC-Tiger method improves data security in real-time intrusion detection systems by balancing speed and entropy while generating hash codes. Because of its rapid processing, latency is decreased, and hash code security is enhanced, increasing its resistance to manipulation or attack. The tiger hashing algorithm offers quick and efficient generation of hash codes for large amounts of data. It uses Euclidean distance for large data sets, but it's not scale-invariant, potentially resulting in the same hash code across different texts. The 64-bit hash code generating process known as the PEC-Tiger algorithm includes input preparation, initialization, compression function, and finalization. It generates a 192-bit hash, optimized for 64-bit computers and employs strong cryptographic approaches for security by utilizing bitwise rotations, linear transformations, XOR operations, nonlinear S-boxes, and modular arithmetic. So, instead of Euclidean distance, the proposed framework utilized Polychoric Entropy Correlation, eliminating the risk of the same hash code generation for different texts. The polychoric entropy correlation coefficient is a tool that measures the strength of the relationship between latent variables while considering data entropy ($\wp$). For safe IDS operations, it guarantees strong, collision-resistant hashing and improves hash generation accuracy. When used with LJS-POA, hash code creation enhances load balancing in distributed systems. By ensuring consistent, scalable, and uniform job allocation, it minimizes overloading and maximizes search space. LJS-POA uses hashed distribution for resource usage, dynamic system changes, and task distribution. The process of the PEC-Tiger approach is illustrated as

Firstly, the data from$\eta\vartheta$ are initialized with predefined constants (a, b, c) and then, $\Re\beta$ are divided into b number of blocks, which are represented as,

$$\Re\beta_B = (\Re\beta_1, \Re\beta_2, \Re\beta_3, \cdots, \Re\beta_b); B = 1 to b \tag{2}$$

Subsequently, the Polychoric entropy correlation coefficient $\left(f^{\wp}{}_{PEC}\right)$ for $i_{th}$and j_{th}block is evaluated to ensure accurate hashing, which is explained as,

$$f^{\wp}{}_{PEC} = \sum\nolimits_{i,j} E(\Re\beta_i) \cdot E\left(\Re\beta_j\right) \cdot f_{PCC}\left(\Re\beta_i, \Re\beta_j\right) \tag{3}$$

$$f_{PCC} = \frac{\sinh^{-1}\left(\rho_{per}\right)}{\pi} \tag{4}$$

Where, $E(\Re\beta)$is the entropy of each$\Re\beta_B$, $f_{PCC}$is the polychoric correlation coefficient, and $\rho_{per}$is the Pearson correlation coefficient. The unpredictability and uncertainty of the input data during the hashing process are measured by the Entropy Representation ($\Re$). More secure hash codes result from varied and non-redundant material, which is indicated by high entropy. This is determined by Shannon's Entropy Formula, which guarantees a consistent distribution of hash results. To ensure safe data, hashing is combined with it.

Now, the compression function $\left(f_{COM}\right)$ is applied to each$\Re\beta_B$ and the hashed data$(\psi_{hash})$is illustrated as,

$$\psi_{hash}(\Re\beta_B) = f_{COM}(\Re\beta_B) \tag{5}$$

Then, $a, b,$ care updated based on$f_{COM}(\Re\beta_B)$, which is depicted as,

$$\left(a_{up}, b_{up}, c_{up}\right) = \left(a \oplus f_{COM}(\Re\beta_B), b \oplus f_{COM}(\Re\beta_B), c \oplus f_{COM}(\Re\beta_B)\right) \tag{6}$$

Where, $a_{up}, b_{up}, c_{up}$are the updated constants.

A big prime integer for uniform distribution is used to define the unique identifiers inside the dataset and constants from the feature set to construct the hash (hh). Where m is the size of the hash table, and x is the data element, the hash is produced using the hash function. The pseudo-random generation and collision resistance criteria were used to choose these constants. Finally, a_{up}, b_{up}, c_{up} are combined to derive a unique hash, which is demonstrated as,

$$\psi_{hash}{}^{Final} = \left\{a_{up}, b_{up}, c_{up}\right\} \tag{7}$$

Where, $\psi_{hash}{}^{Final}$is the final hashed data. Based on$\psi_{hash}{}^{Final}$, the partial digital signature is created through login. Creating a digital signature by applying a digital signature algorithm to a hashed copy of the original material is a partial digital signature. The algorithm determines mathematical operations and cryptographic keys. It is encoded before being sent or stored to guarantee the final signature's legitimacy and integrity. Once a login is successful, the user can access the data for further use. The hashing process converts input data into a fixed-length hash code using the constants hh, hh, and h. To guarantee that different inputs translate to other outputs, these constants—a, b, o, α, and c—control the mapping from input space to hash space. Constraint-based randomization is used to select constants.

3.3. Partial digital signature creation

From$\psi_{hash}{}^{Final}$, the partial digital signature is created by using the ELCG-DSA technique. The ELCG-DSA partial digital signature creation procedure reduces assaults like replay attacks and signature forging by producing distinct, unexpected values that guarantee data integrity and authenticity. This procedure ensures cryptographic security, verifies the authenticity of the contents, and increases efficiency by signing only a piece of the message. The DSA is a secure and efficient signature algorithm, but its limited key lengths make it less suitable for specific applications. Hence, to overcome this, the Exponential Linear Congruential Generator function is applied, which offers extended periods and improved randomness, reducing the correlation between generated keys. The Exponential Linear Congruential Generator (ELCG) provides a more unexpected key generation technique, which enhances the security and unpredictability of the Digital Signature Algorithm (DSA). Increasing randomness, preventing weaknesses like key reuse, and making digital signatures more difficult to anticipate strengthens cryptography overall. The process of ELCG-DSA is illustrated as,

Key Generation: From the key generation center, a private key$\left(\kappa_{pri}\right)$, and its corresponding public key$\left(\kappa_{pub}\right)$ are created by using the exponential linear congruential generator function, which is elaborated as,

$$\kappa_{pub} = \left(\beta_a{}^{\kappa_{pri}} \exp\left(\kappa_{pri}\right) + \beta_b{}^{\kappa_{pri}}\right) \cdot \mathrm{mod}\left(\kappa_{pri}\right) \tag{8}$$

Where, $\beta_a$and $\beta_b$are the base parameters.

Signature Generation: After key generation, the signatures$\left(\xi_{Sig}\right)$ are generated using the following representation.

$$\xi_{Sig} = \left(\beta_{base}{}^{r} \mathrm{mod}\left(\kappa_{pri}\right)\right) \mathrm{mod}\left(\kappa_{pub}\right) \tag{9}$$

Then, ξ_{Sig} is the signatures.

Signature Verification: Finally, the recipient applies the signature verification function$\left(f_{Sig}{}^{Ver}\right)$ to the received message $(\mathrm{Re}(\eta\vartheta))$and ξ_{Sig} using κ_{pub}. This can be denoted as,

$$\xi_{Sig}{}^{Ver} = f_{Sig}{}^{Ver}\left(\mathrm{Re}(\eta\vartheta), \xi_{Sig}(\eta\vartheta)\right) \tag{10}$$

Where, $\xi_{Sig}{}^{Ver}$is the verified signature.

3.4. Data security

After signature creation, $\eta\vartheta$are sensed through login to ensure data security, which uses the MMCC technique. Montgomery Modulo Curve Cryptography (MMCC) uses ECC to generate more secure cryptographic keys, improving private key creation in an IDS system. Particularly in delicate settings like IIoT, this method lowers the chances of key compromise and illegal access, improving system security overall. ECC enhances security and makes it well-suited for modern digital security needs; however, the private key generation using random values causes problems in private key generation. Hence, a Montgomery Modulo Curve is utilized to overcome this by replacing the elliptic curve, which offers efficient and secure generation of private keys. The process of MMCC is illustrated as,

Firstly, for a cryptographic operation, the Montgomery modulo curve is utilized, which is depicted as,

$$C_y E_b{}^2 = E_a{}^3 + C_x E_a{}^2 + E_a \tag{11}$$

Where E_a and E_b are the variables denoting the coordinates of the Montgomery modulo curve,C_x and C_y are constants.

Then,κ_{pri} is chosen from the key generation centre, while $\kappa_{pub}$is derived from$\eta\vartheta$ using the following representation.

$$\kappa_{\mathrm{pri}} = \kappa_{\mathrm{pub}} \cdot \delta_{SP} \tag{12}$$

Where δ_{SP} is the starting point of the curve.

Now, $\eta\vartheta$is encrypted usingκ_{pub}. Then, the resulting cipher text($\xi\mathscr{C}$) is illustrated as

$$\xi\mathscr{C} = \left(r_{sc} \cdot \delta_{SP} + \kappa_{pub}\right) \tag{13}$$

Where, $r_{sc}$is the random scalar value. Finally, the secured data($\xi\mathscr{C}$)are is processed for load balancing through data verification in the edge server, and the data verification process is represented as follows. The use of TRNGs or CSPRNGs to generate random scalar values is necessary to keep encryption unpredictable. They provide ephemeral keys, distinct encryption settings, unique ciphertext, and defence against side-channel attacks. They are also essential for secure protocols because they defend cryptographic operations, avoid key reuse, and withstand cryptanalysis. Authenticated encryption, digital signatures, MACs, and cryptographic algorithms like SHA-256 are some methods used to guarantee data integrity. Checksums, CRCs, and ECC are error detection methods that fix transmission or storage problems, while secure environments like TEEs and homomorphic encryption separate sensitive data. Access is restricted and integrity violations are detected by monitoring and access controls.

3.5. Data verification

The data verification is carried out in the edge server by generating hash codes using PEC-Tiger, as explained in section 3.2. The generated hash codes are then compared with the corresponding codes stored in IPFS to verify their consistency. Polychoric Entropy Correlation (PEC) increases the quality and efficiency of data representation in the Inter-Planetary File System (IPFS), which enhances hash code production. It improves data integrity and makes hash codes more resilient to collisions, guaranteeing safe storage and retrieval of big information. If they match, the edge server notifies the key generation centre to create fully digital signatures using ELCG-DSA, as explained in section 3.3. The edge server is essential for data verification since it guarantees the authenticity and integrity of the data. It authenticates data at the source, detecting abnormalities or tampering early, improving scalability and decreasing latency. It also carries out preparatory processing utilizing lightweight calculations like hashing or encryption. Then, the fully digital signatures access the data from the edge serve $\chi^{\psi}$for load balancing.

3.6. Load balancing

Here, load balancing of$\chi^{\psi}$is carried out by using the proposed LJS-POA. POA has faster convergence speed and simple calculations, enabling wider solution space exploration. Using hash codes, the LJS-POA ensures equitable distribution by optimizing job distribution. Employing stochastic search and pelican-inspired behaviour to reallocate jobs dynamically maximizes resource utilization and adjusts to real-time changes, guaranteeing scalability and performance in dispersed systems. Data security procedures protect the confidentiality and integrity of data, while load sharing distributes duties among nodes to improve system efficiency. Balancing scalability and protection, ensuring safe synchronization, and mitigating vulnerabilities promote dependability and trust within a secure framework. However, it faces issues in random position updation, which leads to local optima. So Luus-Jaakola sequence is included to overcome this issue, which enhances the exploration phase by systematically sampling the solution space. The process of LJS-POA is illustrated as follows,

Firstly, the population or the collection of individuals in $\chi^{\psi}$is randomly initialized using the following equation.

$$\chi^{\psi}{}_{i,j} = \hbar^{low}{}_j + r * \left(\hbar^{upp}{}_j - \hbar^{low}{}_j\right) \quad i = 1toN, j = 1toM \tag{14}$$

Where Mindicates the problematic variables, $\hbar^{upp}{}_j$ and $\hbar^{low}{}_j$depicts the upper and lower boundary of j_{th} problematic data, and r illustrates the random variable.

Here, the generation of random perturbation (χ^{per}) is done by using the Luus-Jaakola sequence$\left(\alpha^{LJS}\right)$, which is illustrated as,

$$\alpha^{LJS}(\chi^{per}) = \chi^{per} + \Delta \cdot (2 \cdot r - 1) \tag{15}$$

$$\alpha^{LJS}(\chi^{per}{}_p) = \left\{\chi^{per}{}_1, \chi^{per}{}_2, \chi^{per}{}_3, \cdots, \chi^{per}{}_p\right\}; P = 1 \rightarrow p \tag{16}$$

Where,p is the number of perturbations, and $\Delta$is the size of perturbations. Next, the new position of i_{th} candidate solution$\left(\Im\left(\chi^{\psi}{}_{i,j}\right)\right)$is signified as,

$$\Im\left(\chi^{\psi}{}_{i,j}\right) = \begin{cases} \chi^{\psi}{}_{i,j} + r \cdot \left(\partial_j - R\chi^{\psi}{}_{i,j}\right) + \chi^{per}{}_i, OF_\partial < OF_i \\ \chi^{\psi}{}_{i,j} - r \cdot \left(\partial_j - R\chi^{\psi}{}_{i,j}\right) + \chi^{per}{}_i, otherwise \end{cases} \tag{17}$$

$$OF(\chi^{\psi}) = \sum_{i=1}^{N} \left(\chi^{\psi}{}_{i,1}{}^2 + \chi^{\psi}{}_{i,2}{}^2 + \cdots + \chi^{\psi}{}_{i,M}{}^2\right) \tag{18}$$

Where, $\partial_j$is the position of the target, $R$is the parameter that can be either 1 or 2, andOF is the objective function value.

Then, $\chi^{\psi}{}_i$is updated based on$\Im\left(\chi^{\psi}{}_{i,j}\right)$, which is outlined as,

$$\chi^{\psi}{}_i = \begin{cases} \Im(\chi^{\psi}{}_i), \Im(OF_i) < OF_i \\ \chi^{\psi}{}_i, otherwise \end{cases} \tag{19}$$

Where, $\Im(OF_i)$is the objective function value based on the current position.

Now, the new position$\left(\Im_{new}\left(\chi^{\psi}{}_{i,j}\right)\right)$ is evaluated based on the search for optimal solutions; then, the behaviour of each $\chi^{\psi}{}_{i,j}$is represented as,

$$\Im_{new}\left(\chi^{\psi}{}_{i,j}\right) = \chi^{\psi}{}_{i,j} + C \cdot \left(1 - I/I_{\max}()(2 \cdot r - 1)^{\psi}{}_{i,j}\right) \tag{20}$$

Where, $C$is the constant value, $I$and $I_{\max}$is the current and maximum iteration, respectively.

Finally, $\chi^{\psi}{}_i$ is updated based on$\Im_{new}\left(\chi^{\psi}{}_{i,j}\right)$, which is illustrated as,

$$\chi^{\psi}{}_{i} = \begin{cases} \Im_{new}(\chi^{\psi}{}_{i}), \Im_{new}(OF_i) < OF_i \\ \chi^{\psi}{}_{i}, otherwise \end{cases} \quad (21)$$

Then, the steps are repeated until convergence, and the balanced data is represented by$\sigma\zeta$. The pseudo-code for the proposed LJS-POA is depicted as,

Pseudo-code for the proposed LJS-POA

Input: $\xi\ell$

Output: $\sigma\zeta$

Begin

Initialize $\xi\ell_{i,j}, \chi^{per}, \Im(\xi\ell_{i,j}), \Im_{new}(\xi\ell_{i,j})$, iteration$(I)$, maximum iteration$(I_{max}())$

Set$(I = 1)$

While$(I_{max}())$

For each $\xi\ell$do

Initialize population

$$\xi\ell_{i,j} = \hbar^{low}{}_{j} + r * \left(\hbar^{upp}{}_{j} - \hbar^{low}{}_{j}\right)$$

Generate χ^{per}

$$\chi^{per}{}_{P} = \left\{\chi^{per}{}_{1}, \chi^{per}{}_{2}, \chi^{per}{}_{3}, \cdots, \chi^{per}{}_{p}\right\}$$

Calculate position

$$\Im(\xi\ell_{i,j}) = \begin{cases} \xi\ell_{i,j} + r \cdot \left(\partial_j - R\xi\ell_{i,j}\right) + \chi^{per}{}_{i}, OF_{\partial} < OF_i \\ \xi\ell_{i,j} - r \cdot \left(\partial_j - R\xi\ell_{i,j}\right) + \chi^{per}{}_{i}, otherwise \end{cases}$$

Update position

$$\xi\ell_i = \begin{cases} \Im(\xi\ell_i), \Im(OF_i) < OF_i \\ \xi\ell_i, otherwise \end{cases}$$

Find new position

$$\Im_{new}(\xi\ell_{i,j}) = \xi\ell_{i,j} + C \cdot \left(1 - I/I_{max}\,()(2 \cdot r - 1)_{i,j}\right)$$

Update new position

$$\xi\ell_i = \begin{cases} \Im_{new}(\xi\ell_i), \Im_{new}(OF_i) < OF_i \\ \xi\ell_i, otherwise \end{cases}$$

End For

End While

Return: Balanced data

End

After load balancing, $\sigma\zeta$is processed for testing the data in the IDS training set, in which the IDS training includes, big-data handling, preprocessing, data balancing, feature extraction, feature selection, and classification. Enhancements to the Intrusion Detection System (IDS) include lowering computational cost by removing superfluous features, boosting generalization by lowering the danger of overfitting, and increasing model accuracy by concentrating on useful features. Mutual information criteria or recursive feature elimination (RFE) methods find and retain optimal features.

3.7. Preprocessing

Here, data$(\lambda\vartheta)$ from the dataset are collected, and the big data handling process is conducted using the Hadoop Distributed File System (HDFS). With its ability to enable scalable, dependable, and effective storage and retrieval across several cluster nodes, HDFS is essential for managing big datasets in IIoT contexts. By guaranteeing fault tolerance, high availability, and parallel processing, it makes it possible for intrusion detection systems to effectively handle massive amounts of sensor and device data in real-time. HDFS efficiently stores and manages large volumes of data by providing scalability and high throughput, and the data is depicted asτv. Then,$\tau v$is is preprocessed to enhance the raw sensor data under Missing Value Imputation (MVI), Numeralization, and Normalization to detect and classify intrusions effectively. Feature extraction, standardization, and normalization are used in the pre-processing phase of data analysis. Normalization rescales data values to a specified range, standardization converts features to a mean and standard deviation, and feature extraction chooses the most important

properties. These processes lessen biases, enhance model performance, and speed up training convergence. This is elaborated as follows,

3.8. MVI

Initially, the missing values(τ_{mis}) in$\tau\upsilon$ are identified, which contain Not a Number (NaN), null values, or another placeholder. Then, the nature ($\varepsilon\varsigma$)of $\tau_{mis}$is is determined by whether they are Missing Completely At Random (MCAR), Missing At Random (MAR), or Missing Not At Random (MNAR).

The procedure entails assessing the data distribution and locating missingness patterns (MCAR, MAR, or MNAR). Based on observable or unobserved data, missingness can be categorized as MCAR or MNAR. Imputation methods include k-nearest neighbours and mean/mode imputation. Metrics such as RMSE and sensitivity analysis are used to assess performance. Next, based on$\varepsilon\varsigma$, for numerical, data ifτ_{mis} is MCAR or MAR, the mean ($\vec{\mu}$) and median ($\vec{m}$) imputation is considered, and if MNAR, then Regression imputation $\left(\vec{R}\right)$ is considered. Meanwhile, if $\tau_{mis}$is MCAR or MAR, the mode imputation ($\vec{m}_{mi}$) is considered for categorical data. This can be illustrated as,

$$\vec{\mu} = \sum_{N=1}^{n} \frac{\tau\upsilon_N}{n} \tag{22}$$

$$\vec{m} = \text{median}(\tau\upsilon) \tag{23}$$

$$\vec{R} = re_0 + re_1\tau\upsilon_{N1} + re_2\tau\upsilon_{N2} + \cdots + re_p\tau\upsilon_{pN} + er_N \tag{24}$$

$$\vec{m}_{mi} = \text{mode}(\tau\upsilon) \tag{25}$$

Where, *re*is the regression coefficients, *p* is the number of *re*, and *er* is the error term.

After filling $\tau_{mis}$with calculated$\vec{\mu}$, $\vec{m}$, $\vec{R}$, and$\vec{m}_{mi}$, the process ensures that no biases are introduced for effective imputation of τ_{mis}. Then, the data after MVI is depicted as$\upsilon\varpi$.

Numeralization

Primarily from$\upsilon\varpi$, a unique numerical label(ς_{LB}) of i_{th}category(Cat)is assigned to transform the categorical data, which is depicted as,

$$Cat_C = \{Cat_1, Cat_2, Cat_3, \cdots, Cat_c\}; C = 1 \rightarrow c \tag{26}$$

$$\varsigma_{LB}(Cat_i) = i - 1 \tag{27}$$

Where, c is the number of categories in $\upsilon\varpi$.

Then, the binary dummy variables are created using a one-hot encoding function(f_{ohe}) for each category, which is illustrated as,

$$f_{ohe}(\upsilon\varpi_i, Cat_i) = \begin{cases} 1; if\, \upsilon\varpi_i \in Cat_i \\ 0; otherwise \end{cases} \tag{28}$$

Finally, based on the order ofς_{LB}, binary dummy variables are assigned toCat. Then, the numeralized data is represented asϑ_{num}.

Normalization

Here, $\vartheta_{num}$are normalized to ensure optimal data utilization for predictive maintenance and optimization endeavours in IIoT networks. This process is illustrated as,

Initially, the numerical columns(nc) from $\vartheta_{num}$are identified, and then, the scaling functions are applied to adjust eachnc inϑ_{num}.

Then, the scaled data($sca(\vartheta_{num})$) is verified whether the changes cause problems. Finally,$sca(\vartheta_{num})$are considered preprocessed data and ready for further analysis. After that, $sca(\vartheta_{num})$are processed for data balancing.

3.9. Data balancing

Here, $sca(\vartheta_{num})$are balanced by using SMOTE. This technique addresses the class imbalance problem by generating synthetic samples for minority classes. The IDS framework's SMOTE function aids in balancing unbalanced datasets, especially when there are fewer intrusion incidents than usual. It produces fictitious data points for the minority class, increasing the intrusion detection accuracy of the IDS model and enabling the identification of infrequent and disregarded assaults. This process is demonstrated as,

Primarily, the minority class set($\curlywedge\varsigma_{\min}()$) of $sca(\vartheta_{num})$ is selected, followed by the computation of the distances between each data in $\curlywedge\varsigma_{\min}$ to identify its k-nearest neighbours.

Then, the sampling rate(sr) is selected for each$n \in mc$ according to the imbalanced proportion of$sca(\vartheta_{num})$.

$$\curlywedge\varsigma\{n_1, n_2, n_3, \cdots, n_{sr}\}_{\min} \tag{29}$$

Where, $\widehat{\curlywedge\varsigma}_{\min}$is the set of selected$sr$.

Next, the synthetic instances($\widehat{n}$) are generated for each$n_\nu \in \widehat{\curlywedge\varsigma}_{\min}$,

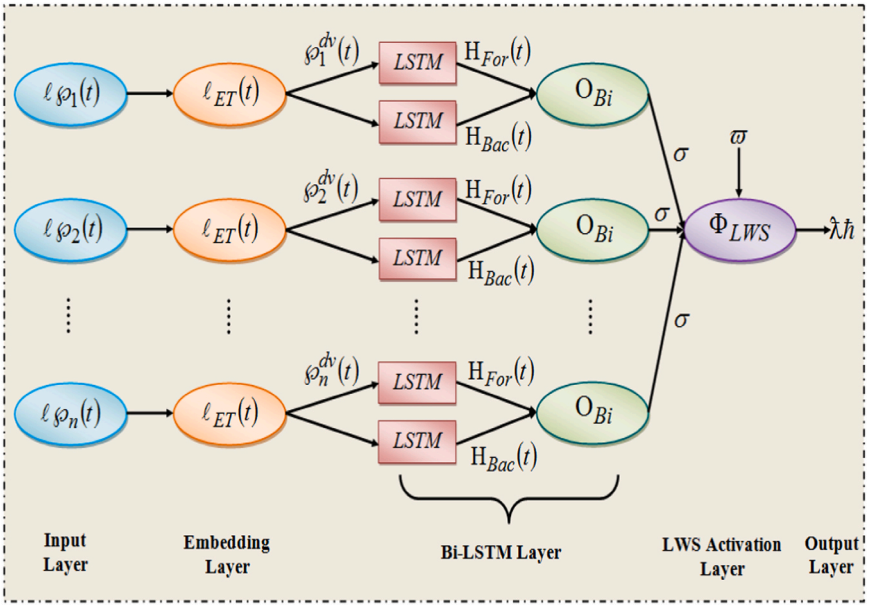


Fig. 2. Classifier diagram for the proposed LWS-BiOLSTM.

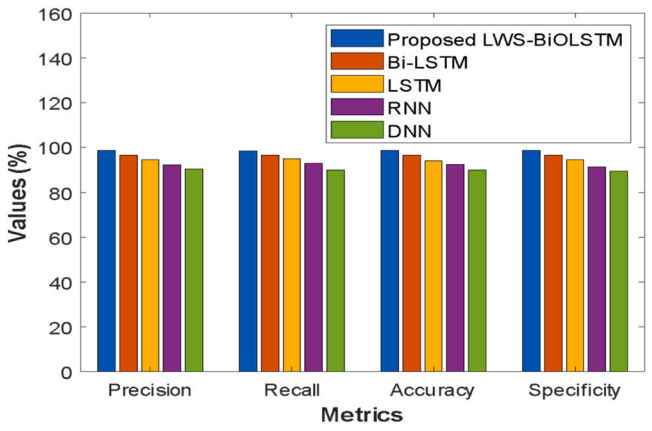


Fig. 3. Performance validation regarding precision, recall, accuracy, and specificity.

Table 1
Comparison by FPR and FNR.

Techniques	FPR	FNR
Proposed LWS-BiOLSTM	1.5124	1.5421
Bi-LSTM	3.6632	3.7125
LSTM	5.1058	4.9452
RNN	8.4107	6.6307
DNN	10.6307	9.5245

using the following representation,

$$\widehat{n} = r \times |n - n_{\nu}| + n; \nu = 1 to sr \tag{30}$$

Lastly, the balanced class labels($\xi\hbar$) are formed by combining $n$and$\widehat{n}$.

3.10. Feature selection

From$\xi\hbar$, features like destination port, forward packet length maximum, and so on are extracted for intrusion detection and classification. Then, the LJS-POA is applied to select features from extracted features, and the process is demonstrated in section 3.6. Moreover, the selected features($\ell\wp$), such as Destination port, flow duration, flow byte, and so on, are the final optimal solutions.

3.11. Classification

Here, $\ell\wp$are processed for the detection and classification of intrusions by using the LWS-BiOLSTM. Bi-LSTM processes input sequences

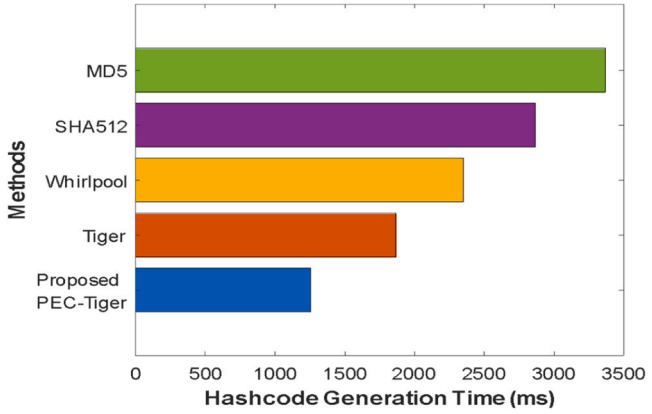


Fig. 4. Hashcode generation time analysis.

in both forward and backward directions, reducing overfitting; however, it requires more memory, and the performance can be sensitive to random weight initializations. Limited feature sets and models that capture noise are potential causes of overfitting. Examples of mitigation tactics include data augmentation, feature selection methods, cross-validation, and regularization. Recursive Feature Elimination is one feature selection strategy; cross-validation verifies performance across various subsets, and regularization penalizes complicated models. To reduce memory usage, Log Wave Sigmoid activation function and once initializer is used. Then the classifier diagram is depicted in Fig. 2.

Input Layer: Initially, each$\ell\wp$ is represented as a vector at the time step t. Then, the input sequence is expressed as,

$$\ell\wp_{\boldsymbol{N}}(\boldsymbol{t}) = (\ell\wp_{\boldsymbol{1}}(\boldsymbol{t}), \ell\wp_{\boldsymbol{2}}(\boldsymbol{t}), \ell\wp_{\boldsymbol{3}}(\boldsymbol{t}), \cdots, \ell\wp_{\boldsymbol{n}}(\boldsymbol{t})) \tag{31}$$

Embedding Layer: Next, $\ell\wp_N(t)$are transformed into dense vector representation $(\wp_i^{dv})$ by capturing the semantic meaning of $\ell\wp_i$, which is illustrated as,

$$\ell_{ET}(t) = \ell\wp_i(t) \rightarrow \wp_i^{dv}(t) \tag{32}$$

Where, ℓ_{ET}indicates the embedding transformation function.

Bi-LSTM Layer: This layer consists of two LSTM layers, one processing in the forward direction(H_{For}) and the other in the backward direction(H_{Bac}). This can be depicted as,

$$H_{For}(t) = \boldsymbol{LSTM}\left(\wp^{dv}(t), H_{For}(t-1)\right) \tag{33}$$

$$H_{Bac}(t) = \boldsymbol{LSTM}\left(\wp^{dv}(t), H_{Bac}(t-1)\right) \tag{34}$$

Where, $t-1$represents the previous time steps.

Output Layer: Next, $H_{For}$and $H_{Bac}$are combined, representing the output from the Bi-LSTM layer(O_{Bi}).

$$O_{Bi} = [H_{For}(t); H_{Bac}(t)] \tag{35}$$

Finally, O_{Bi}passed through the fully connected layer followed by the log wave sigmoid activation function (Φ_{LWS}), which is represented as,

$$\lambda\hbar = \Phi_{\mathrm{LWS}}(\sigma \cdot \mathrm{O_{Bi}} + \varpi) \tag{36}$$

$$\Phi_{\mathrm{LWS}}(\mathrm{O_{Bi}}) = \frac{1}{1+\mathrm{e}^{-\mathrm{O_{Bi}}}}\left(1+\mathrm{O_{Bi}}^{2}\right)\mathrm{e}^{-\mathrm{O_{Bi}}^{2}} \tag{37}$$

Where $\lambda\hbar$illustrates the final output from the output layer,$\sigma$and$\varpi$is the weight and bias values, respectively. Here, ϖ is initialized as 1 and then, $\lambda\hbar$classifies the IIoT data as, whether it is attacked or not and the various types of attacks such as Brute, Started Query Language (SQL) Injection, Cross-site Scripting (XSS), Distributed Denial-of-Service (DDoS), PortScan, Bot, Infiltration, File Transfer Protocol (FTP), and Secure Shell (SSH), which are illustrated as,

$$\lambda\hbar = \{Brute, SQL\ injection, XSS, DDoS, PortScan, Bot, Infiltration, FTP, SSH\} \tag{38}$$

Attacked information is blocked, while non-attacked data is stockpiled in the cloud. Then, the cloud validates the integrity of the data using a digital signature. Once verified, the authenticated information is securely transmitted to the ICS for storage. This ensures that only authenticated and unaltered data is stored within the ICS, enhancing integrity and security in the IIoT environment. The simplified LWS-BiOLSTM pseudo-code simplifies difficult stages for novices by dividing data into smaller, overlapping chunks, processing each chunk via a Bi-LSTM model, and then combining the results into a single representation for classification or prediction. The pseudo-code for the proposed LWS-BiOLSTM is illustrated as,

Pseudo-code for the proposed LWS-BiOLSTM

Input: $\ell\wp$
Output: $\lambda\hbar$
Begin
Initialize $\ell\wp_N(t), \ell_{ET}(t), O_{Bi}$, iteration$(I)$, and maximum iteration$(I_{max}())$
Set$(I = 1)$
While$(I_{max}())$
For each $\ell\wp$do
Compute input sequences
$$\ell\wp_N(t) = \left(\ell\wp_1(t), \ell\wp_2(t), \ell\wp_3(t), \cdots, \ell\wp_n(t)\right)$$
Perform embedding function
$$\ell_{ET}(t) = \ell\wp_i(t) \rightarrow \wp_i^{dv}(t)$$
Estimate forward LSTM
$$H_{For}(t) = LSTM\left(\wp^{dv}(t), H_{For}(t-1)\right)$$
Evaluate backward LSTM
$$H_{Bac}(t) = LSTM\left(\wp^{dv}(t), H_{Bac}(t-1)\right)$$
Combine both LSTM output
$$O_{Bi} = [H_{For}(t); H_{Bac}(t)]$$
Apply log wave sigmoid activation function
$$\Phi_{LWS}(O_{Bi}) = \frac{1}{1+e^{-O_{Bi}}} - log(1 + e^{-O_{Bi}})$$
Examine output layer
$$\lambda\hbar = \Phi_{LWS}(\sigma \cdot O_{Bi} + \varpi)$$
End For
End While
Return: $\lambda\hbar = \{Brute, SQLInjection, XSS, DDoS, PortScan, Bot, Infitration, FTP, SSH\}$
End

4. Result and discussion

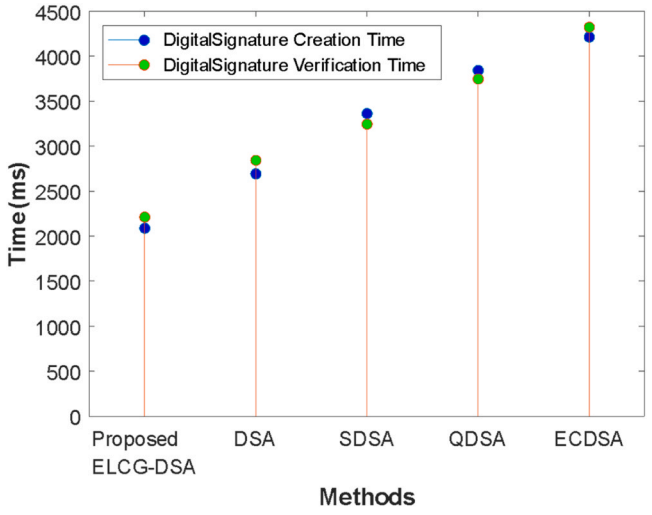


Fig. 5. Graphical representation of the proposed model and existing techniques.

Here, the proposed research's performance and comparative analysis are conducted to prove the model's reliability and trustworthiness. The proposed model is executed on the working platform of PYTHON. Python 3.8 + is the environment, while Jupyter Notebook/VS Code is the IDE. Pandas, NumPy, Scikit-learn, TensorFlow, PyTorch, Matplotlib, Seaborn, SciPy, and statsmodels are important libraries. Consistent library versions are one of the reproducibility measurements.

Table 2
Performance assessment of the proposed model.

Techniques	Encryption Time (ms)	Decryption Time (ms)	Security Level (%)
Proposed MMCC	1267	1325	98.71
ECC	1862	1748	96.36
RSA	2347	2236	94.19
ElGamal	2796	2841	92.14
DES	3154	3364	90.26

4.1. Dataset description

The proposed framework was assessed by utilizing the NSL-KDD dataset. In this work, the NSL-KDD is used to perform attack detection. The NSL-KDD contains 125973 records containing important information about network security, information security, and cyber-attacks. Likewise, from the whole data, 80 % of the data is utilized to perform training, whereas the remaining 20 % is allocated for testing purposes.

4.2. Performance analysis

Here, the performance analysis of the proposed model and prevailing techniques is conducted to demonstrate the model's effectiveness.

Fig. 3 demonstrates the performance validation of the proposed model and existing techniques regarding precision, recall, accuracy, and specificity. Here, the proposed LWS-BiOLSTM obtained a high precision, recall, accuracy, and specificity of 98.78 %, 98.52 %, 98.78 %, and 98.63 %, respectively. However, the existing techniques such as Bi-

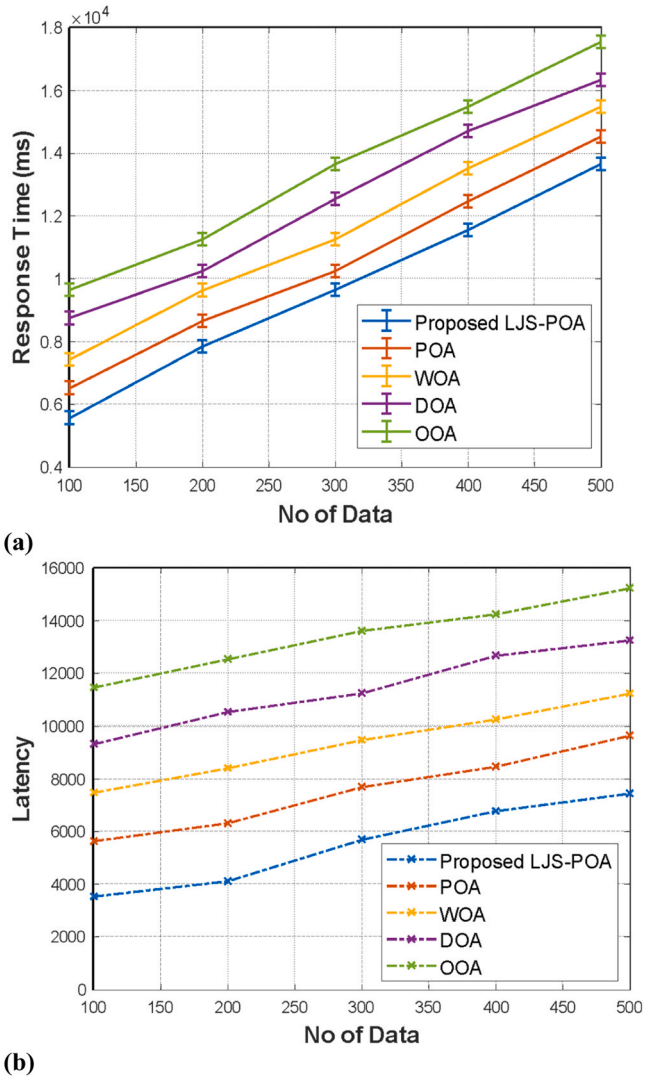


Fig. 6. Graphical representation regarding (a) response time and (b) latency.

Table 3
Comparative assessment regarding load balancing.

Technique/number of data	100	200	300	400	500
Proposed LJS-POA	1875	2786	3865	4794	5896
POA	3845	4785	5698	6895	7845
WOA	5884	6847	7481	8748	9654
DOA	7784	8475	9856	10658	11547
OOA	9912	10254	11574	12547	13659

Table 4
Comparative analysis.

Author's name	Methods	Accuracy (%)	Recall (%)
Proposed model	LWS-BiOLSTM	98.78	98.52
Li et al., [14]	Multi-CNN	81.33	56.07
Awotunde et al., [3]	Ensemble classifier	93.4	92.8
Long et al., [15]	RCLN	97.2	96.8
Gaber et al., [7]	PSO and BA	97.83	97.7
Rashid et al., [18]	FL	93.5	93

LSTM, LSTM, RNN, and DN obtained a low average precision, recall, accuracy, and specificity of 93.51 %, 93.74 %, 93.36 %, and 93.05 %, accordingly. The proposed model employed the Log Wave Sigmoid activation function to reduce memory usage; thus, the model's effectiveness was shown. The Log Wave function is superior to ReLU or Tanh regarding gradient flow, dynamic range, stability, and flexibility. It helps deep networks overcome vanishing gradient issues, prevent harsh saturation points, smooth out input transitions, and better adjust to different input scales. Unlike traditional functions, Log Wave preserves more subtle fluctuations and stays clear of "dying neurons".

Table 1 compares the False Positive Rate (FPR) and False Negative Rate (FNR) of the proposed LWS-BiOLSTM and prevailing techniques. The proposed model obtained a low FPR and FNR of 1.5124 and 1.5421, respectively. FPR and FNR computations should be precisely defined using their respective formulae to guarantee repeatability. Calculations using sample data can be explained using confusion matrices. However, Python snippets are crucial for automated computations, dataset descriptions, preprocessing, and split ratios. Transparency and repeatability may also be improved by sharing Python environment information and using random seeds to provide consistent outcomes. However, the existing models, such as Bi-LSTM, LSTM, RNN, and DNN, obtained a high FPR of 3.6632, 5.1058, 8.4107, and 10.6307, respectively. Likewise, the existing models obtained high FNR values. Thus, the outcomes displayed that the proposed model is better at detecting intrusions due to using the Log Wave Sigmoid activation function and once initializer.

Hashcode generation time analysis of the proposed model and conventional methods such as Tiger, whirlpool, Secure Hashing Algorithm 512 (SHA512), and Message Digest Algorithm 5 (MD5) are shown in Fig. 4. PEC-Tiger is a safe hashing algorithm that performs better in settings requiring accuracy and speed by striking a balance between speed and entropy representation. Accuracy is increased by its polychoric entropy correlation, which guarantees nonlinear connections. PEC-Tiger is perfect for real-time applications like intrusion detection systems (IDS) since it requires less computing power than MD5. Here, the proposed PEC-Tiger algorithm took a less hashcode generation time of 1254 ms. However, the conventional techniques took a high average hashcode generation time of 2610 ms. Thus, the proposed model used Polychoric Entropy Correlation for efficient hashcode generation.

Fig. 5 shows the graphical demonstration of the proposed system and existing techniques such as DSA, Simple DSA (SDSA), Qualified DSA (QDSA), and Elliptic Curve DSA (ECDSA) concerning digital signature creation time as well as digital signature verification time. Here, the proposed methodology obtained better outcomes than the existing techniques. The proposed ELCG-DSA achieved a low digital signature creation time and digital signature verification time of 2145 ms and 2214 ms, respectively. However, the existing SDSA and QDSA attained a high digital signature creation time of 3365 ms and 3845 ms, accordingly. Likewise, the remaining methods attained high digital signature creation and verification time. Here, the proposed model used the Exponential Linear Congruential Generator function, effectively reducing the correlation between generated keys; thus, the model's effectiveness is proved. Hash functions, digital signatures, key management, symmetric, asymmetric, hybrid, and more sophisticated approaches like homomorphic and post-quantum cryptography are all essential for data security, secrecy, integrity, and authentication. They provide strong and dependable security in digital systems by protecting sensitive data, adhering to compliance regulations, and thwarting cyberattacks.

Performance assessment of the proposed framework and conventional techniques is shown in Table 2. The proposed MMCC obtained a low encryption and decryption time of 1267 ms and 1325 ms, respectively. Also, the proposed MMCC achieved a security level of 98.71 %, higher than the conventional techniques. However, the traditional methods attained a high mean encryption and decryption time of 2539.75 ms and 2547.25 ms, accordingly. Also, the traditional techniques attained a low-security level. Here, the Montgomery Modulo Curve is modified with ECC to provide a secure generation of private keys.

Fig. 6(a) and (b) display the graphical evaluation of the proposed approach and the prevailing techniques regarding response time and latency. Here, the proposed model obtained a low response time and latency of 7423 ms and 5698 for 300 data. However, the prevailing Whale Optimization Algorithm (WOA) attained a high response time and latency of 11247 ms and 9481 for 300 numbers of data. Likewise, other prevailing methods obtained a high response time and latency. Here, the proposed model excellently performs load balancing with the help of the Luus-Jaakola sequence with POA.

Table 3 displays the comparative assessment of the proposed model and prevailing methods such as POA, WOA, Dingo Optimization Algorithm (DOA), and Osprey Optimization Algorithm (OOA) regarding load balancing. Here, the proposed model had a load balancing of 5896 for 500 numbers of data. But, the prevailing methods had a load balancing of 7845, 9654, 11547, and 13659, accordingly, for 500 numbers of data. Thus, the proposed model is superior to the prevailing methodologies due to using the Luus-Jaakola sequence.

4.3. Comparative analysis

This section compares the proposed model and related works to prove its reliability.

Table 4 shows the comparative evaluation of the proposed model and related works regarding accuracy and recall. Here, the proposed model obtained a high precision and recall of 98.78 % and 98.52 % due to using the Log Wave Sigmoid activation function and once initializer. Synchronization protocols, database restrictions, redundancy, and immutable logs preserve consistency, while strict procedures like validation, error detection, and audits guarantee data correctness and consistency. Trust is developed by open and honest management, safe communication, and stringent access control. However, the existing Multi Convolutional Neural Network (Multi-CNN) and Ensemble classifier obtained low accuracy of 81.33 % and 93.4 %. Likewise, the existing Regularized Cross-layer Ladder Network (RCLN) Particle Swarm Optimization (PSO) and Bat Algorithm (BA) obtained a low recall of 96.8 % and 97.7 %, respectively. Likewise, the existing Federated Learning (FL) attained low accuracy and recall values. Thus, the outcomes proved that the proposed model is better at detecting intrusions.

5. Conclusion

This paper presents an efficient IDS for IIoT utilizing ELCG-DSA and LWS-BiOLSTM with blockchain. Here, the proposed research was implemented by using the NSL-KDD dataset. The proposed LWS-BiOLSTM achieved high accuracy, precision, and recall of 98.78 %, 98.78 %, and 98.52 %, respectively, in detecting intrusions, which proved the high effectiveness of the model. Likewise, the proposed PEC-Tiger took less than 1254 ms for hashcode generation, less than the existing techniques. Also, the proposed MMCC achieved a high-security level of 98.71 %, demonstrating the model's high level of security. Similarly, the proposed LJS-POA has a low response time of 9654 ms for 500 data. Thus, the experimental outcomes proved that the proposed methodology was superior in detecting the intrusion.

CRediT authorship contribution statement

Kamruzzaman M M: Writing – review & editing. **Basani Dinesh Kumar Reddy:** Methodology. **Grandhi Sri Harsha:** Validation. **Gudivaka Rajya Lakshmi:** Data curation. **Gudivaka Raj Kumar:** Formal analysis. **Gudivaka Basava Ramanjaneyulu:** Writing – original draft, Supervision. **Murugesan Sundarapandian:** Resources, Software.

Authors' Contributions

All authors have made equal contributions to this article.

Ethics approval

Not applicable.

Funding Statement

The authors did not receive any funding.

Permission to reproduce material from other sources

Yes, you can reproduce.

Clinical trial registration

We have not harmed any human person with our research data collection, which was gathered from an already-published article

Future Recommendation

However, the proposed framework only detected intrusions before storing the data in the cloud. It also failed to concentrate on attacks that happened at the bridge gateway level. This framework will be enhanced in the future by identifying intrusions at the bridge gateway level, such as switches, routers, and hubs.

Dataset link

https://www.kaggle.com/datasets/hassan06/nslkdd

Author statement

All authors made equal contributions to this work. Together, they designed the framework, analyzed performance, validated results, gathered necessary information, provided software, conducted critical reviews and managed the process.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] S. Alosaimi, S.M. Almutairi, An intrusion detection system using BoT-IoT, Appl. Sci. (Switz.) 13 (9) (2023) 1–15, https://doi.org/10.3390/app13095427.
[2] H.C. Altunay, Z. Albayrak, A hybrid CNN + LSTM-based intrusion detection system for industrial IoT networks, Eng. Sci. Technol. Int. J. 38 (2023) 1–13, https://doi.org/10.1016/j.jestch.2022.101322.
[3] J.B. Awotunde, S.O. Folorunso, A.L. Imoize, J.O. Odunuga, C.C. Lee, C.T. Li, D. T. Do, An ensemble tree-based model for intrusion detection in industrial internet of things networks, Appl. Sci. (Switz.) 13 (4) (2023) 1–22, https://doi.org/10.3390/app13042479.
[4] N. Dat-Thinh, H. Xuan-Ninh, L. Kim-Hung, MidSiot: a multistage intrusion detection system for internet of things, Wirel. Commun. Mob. Comput. 2022 (2022) 1–15, https://doi.org/10.1155/2022/9173291.
[5] M.V. Devarajan, S. Aluvala, V. Armoogum, S. Sureshkumar, H.T. Manohara, Intrusion detection in industrial internet of things based on recurrent rule-based feature selection, Proc. 2024 Second Int. Conf. Netw. 2024 (2024) 1–4.
[6] V. Ellappan, A. Mahendran, M. Subramanian, J. Jotheeswaran, A.O. Khadidos, A. O. Khadidos, S. Selvarajan, Sliding principal component and dynamic reward reinforcement learning-based IIoT attack detection, Sci. Rep. 13 (1) (2023) 1–17, https://doi.org/10.1038/s41598-023-46746-0.
[7] T. Gaber, J.B. Awotunde, S.O. Folorunso, S.A. Ajagbe, E. Eldesouky, Industrial Internet of Things Intrusion Detection Method Using Machine Learning and Optimization Techniques, Wirel. Commun. Mob. Comput. (2023) 1–15, https://doi.org/10.1155/2023/3939895.
[8] S. Halder, T. Newe, Radio fingerprinting for anomaly detection using federated learning in LoRa-enabled Industrial Internet of Things, Future Gener. Comput. Syst. 143 (2023) 322–336, https://doi.org/10.1016/j.future.2023.01.021.

[9] H. Huang, P. Ye, M. Hu, J. Wu, A multi-point collaborative DDoS defence mechanism for IIoT environment, Digit. Commun. Netw. 9 (2) (2023) 590–601, https://doi.org/10.1016/j.dcan.2022.04.008.
[10] P.L.S. Jayalaxmi, R. Saha, G. Kumar, M. Alazab, M. Conti, X. Cheng, PIGNUS: a deep learning model for IDS in industrial internet-of-things, Comput. Secur. 132 (2023) 1–14, https://doi.org/10.1016/j.cose.2023.103315.
[11] B. Kadiyala, H. Kaur, Secured IoT data sharing through decentralized cultural co-evolutionary optimization and anisotropic random walks with isogeny-based hybrid cryptography, J. Sci. Technol. 6 (6) (2021) 231–245, https://doi.org/10.46243/jst.2021.v06.i06.pp231-245.
[12] S.M. Kasongo, An advanced intrusion detection system for IIoT Based on GA and tree-based algorithms, IEEE Access 9 (2021) 113199–113212, https://doi.org/10.1109/ACCESS.2021.3104113.
[13] P. Kumar, R. Kumar, A. Kumar, A.A. Franklin, S. Garg, S. Singh, Blockchain and deep learning for secure communication in digital twin empowered industrial IoT network, IEEE Trans. Netw. Sci. Eng. 10 (5) (2023) 2802–2813, https://doi.org/10.1109/TNSE.2022.3191601.
[14] Y. Li, Y. Xu, Z. Liu, H. Hou, Y. Zheng, Y. Xin, Y. Zhao, L. Cui, Robust detection for network intrusion of industrial IoT based on multi-CNN fusion, Meas.: J. Int. Meas. Confed. 154 (2020) 1–27, https://doi.org/10.1016/j.measurement.2019.107450.
[15] J. Long, W. Liang, K.C. Li, Y. Wei, M.D. Marino, A regularized cross-layer ladder network for intrusion detection in industrial internet of things, IEEE Trans. Ind. Inform. 19 (2) (2023) 1747–1755, https://doi.org/10.1109/TII.2022.3204034.
[16] S. Misra, C. Roy, T. Sauter, A. Mukherjee, J. Maiti, Industrial internet of things for safety management applications: a survey, IEEE Access 10 (2022) 83415–83439, https://doi.org/10.1109/ACCESS.2022.3194166.
[17] A. Rahman, M.J. Islam, S.S. Band, G. Muhammad, K. Hasan, P. Tiwari, Towards a blockchain-SDN-based secure architecture for cloud computing in smart industrial IoT, Digit. Commun. Netw. 9 (2) (2023) 411–421, https://doi.org/10.1016/j.dcan.2022.11.003.
[18] M.M. Rashid, S.U. Khan, F. Eusufzai, M.A. Redwan, S.R. Sabuj, M. Elsharief, A federated learning-based approach for improving intrusion detection in industrial internet of things networks, Network 3 (1) (2023) 158–179, https://doi.org/10.3390/network3010008.
[19] G. Rathee, C.A. Kerrache, M.A. Ferrag, A blockchain-based intrusion detection system using viterbi algorithm and indirect trust for IIoT systems, J. Sens. Actuator Netw. 11 (4) (2022) 1–12, https://doi.org/10.3390/jsan11040071.
[20] S. Ruiz-Villafranca, J. Carrillo-Mondejar, J.M. Castelo Gomez, J. Roldan-Gomez, MECInOT: a multi-access edge computing and industrial internet of things emulator for the modelling and study of cybersecurity threats, J. Supercomput. 79 (11) (2023) 1–32, https://doi.org/10.1007/s11227-023-05098-2.
[21] S. Ruiz-Villafranca, J. Roldan-Gomez, J. Carrillo-Mondejar, J.M. Castelo Gomez, J. M. Villalon, A MEC-IIoT intelligent threat detector based on machine learning boosted tree algorithms, Comput. Netw. 233 (2023) 1–15, https://doi.org/10.1016/j.comnet.2023.109868.
[22] A.M.S. Saleh, Blockchain for secure and decentralized artificial intelligence in cybersecurity: a comprehensive review, Block.: Res. Appl. (2024) 1–32, https://doi.org/10.1016/j.bcra.2024.100193.
[23] M.M. Salim, A.K. Comivi, T. Nurbek, H. Park, J.H. Park, A blockchain-enabled secure digital twin framework for early botnet detection in IIoT environment, Sensors 22 (16) (2022) 1–25, https://doi.org/10.3390/s22166133.
[24] S. Selvarajan, G. Srivastava, A.O. Khadidos, A.O. Khadidos, M. Baza, A. Alshehri, J. C.W. Lin, An artificial intelligence lightweight blockchain security model for security and privacy in IIoT systems, J. Cloud Comput. 12 (1) (2023) 1–17, https://doi.org/10.1186/s13677-023-00412-y.
[25] M. Shahjalal, M.M. Islam, M.M. Alam, Y.M. Jang, Implementation of a secure LoRaWAN system for industrial internet of things integrated with IPFS and blockchain, IEEE Syst. J. 16 (4) (2022) 5455–5464, https://doi.org/10.1109/JSYST.2022.3174157.
[26] A.H. Shnain, K. Gattupalli, C. Nalini, P. Alagarsundaram, R. Patil, Faster recurrent convolutional neural network with edge computing-based malware detection in industrial internet of things, Proc. 2024 Int. Conf. Data Sci. Netw. Secur. (2024) 1–4.
[27] J.S. Shyam Mohan, M. Thirunavukkarasu, N. Kumaran, D. Thamaraiselvi, Deep learning with blockchain based cyber security threat intelligence and situational awareness system for intrusion alert prediction, Sustain. Comput.: Inform. Syst. 42 (2024) 100955.
[28] R.K. Singh, M. Dhanaraj, P. Akkas Ali, M. Balaji, Alharbi, Transfer fuzzy learning enabled streebog cryptographic substitution permutation based zero trust security in IIOT, Alex. Eng. J. 81 (2023) 449–459, https://doi.org/10.1016/j.aej.2023.08.084.
[29] N. Swapna, A blockchain-based method for data integrity verification in multi-cloud storage using chain-code and HVT, Int. J. Mod. Electron. Commun. Eng. 12 (1) (2024). ISSN2321-2152.
[30] S. Tharewal, M.W. Ashfaque, S.S. Banu, P. Uma, S.M. Hassen, M. Shabaz, Intrusion detection system for industrial internet of things based on deep reinforcement learning, Wirel. Commun. Mob. Comput. (2022) 1–8, https://doi.org/10.1155/2022/9023719.
[31] D.T. Valivarthi, Blockchain-powered AI-based secure HRM data management: machine learning-driven predictive control and sparse matrix decomposition techniques, Int. J. Mod. Electron. Commun. Eng. 8 (4) (2020). ISSN 2321-2152.
[32] H. Vargas, C. Lozano-Garzon, G.A. Montoya, Y. Donoso, Detection of security attacks in industrial IoT networks: a blockchain and machine learning approach, Electron. (Switz.) 10 (21) (2021) 1–18, https://doi.org/10.3390/electronics10212662.
[33] Y. Wu, H.N. Dai, H. Wang, Convergence of blockchain and edge computing for secure and scalable IIoT critical infrastructures in industry 4.0, IEEE Internet Things J. 8 (4) (2021) 2300–2317, https://doi.org/10.1109/JIOT.2020.3025916.