Privacy Preservation In Iot Networks Through Dehaene–Changeux Model Integrated With Graph Convolutional Layers And Non-Linear Analysis

S P Maniraj[1], Poonam Jagdish Patil[2], Kothapalli Phani Varma[3], Thammisetty Swetha[4], Vijay Kumar Dwivedi[5], Kusuma Rajasekhar[6]

[1]Assistant Professor, Department of Data Science and Business Systems, School of Computing, SRM Institute of Science and Technology Kattankulathur campus, Chennai, Tamilnadu, India, Email: manirajs@Srmist.edu.In
[2]Assistant Professor, Instrumentation Department, Bharati Vidyapeeth College of Engineering, Navi Mumbai, Maharashtra, India, Email: punam.patil@bvcoenm.Edu.In
[3]Assistant Professor, Department of ECE,S.R.K.R Engineering College,Chinaamiram, Andhra Pradesh, India, Email: kothapalliphanivarma@gmail.Com
[4]Assistant Professor, Department of CSE-Data Science, Madanapalle Instituteof Technology & Science, Madanapalle,Andhra Pradesh, India, Email: swethathammisetty7@gmail.Com
[5]Assistant Professor, Department of Mathematics, Vishwavidyalaya Engineering College Ambikapur, Surguja, Chhattisgarh, India, Email: dwivedi.vk69@gmail.com
[6]Professor, Department of ECE, Bonam Venkata Chalamayya Engineering College (A),Odalarevu,Andhra Pradesh, India, Email: raja.bubbly@gmail.com

Received: 25.04.2024 Revised : 26.05.2024 Accepted: 27.05.2024

ABSTRACT
The fast development of the Internet of Things (IoT) has brought privacy preservation in distributed networks more and more of a problem. Sometimes the complex, dynamic interactions defining these systems are not handled by conventional methods of IoT network security. Combining Graph Convolutional Layers (GCL) with Non-Linear Analysis (NLA) and the Dehaene–Changeux Model (DCM), this work offers a novel approach to circumvent these restrictions and hence improve privacy protection in IoT systems. Originally employed to mimic and assess the information flow inside IoT networks, the DCM is changed in this work to show cognitive processes. While GCL assists the model to effectively collect and evaluate the complex network topologies common of IoT contexts, NLA is utilized to identify and lower non-linear dangers to data privacy. The synthetic IoT dataset employed in evaluating the proposed approach consisted in ten thousand nodes and fifty thousand edges, therefore simulating several real-world attack scenarios. Privacy violations were found with a 92.3% accuracy using GCL paired with DCM, 15.4% more accurate than more traditional methods. Moreover, NLA greatly reduced the false positive rate to 3.7%, thereby enhancing the dependability of the privacy-preserving system. The results reveal that combined DCM, GCL, and NLA not only raises IoT network resilience of privacy preservation but also offers a scalable real-time application solution. This work prepares the ground for additional research on cognitive-inspired models for improved security solutions in IoT environments.

Keywords: IoT networks, privacy preservation, Dehaene–Changeux Model, Graph Convolutional Layers, Non-Linear Analysis.

1. INTRODUCTION

The Internet of Things (IoT) has revolutionized many different sectors by letting sensors, devices, and systems to be perfectly integrated. IoT systems create, compile, and transfer vast amounts of sensitive and personal data. Maintaining the privacy and security of this data is quite important considering the increasing frequency of cyberthreats and unauthorized access. IoT networks that respect privacy will help users to keep confidence and obey legal standards. Though basic, conventional security techniques including access control and encryption sometimes fail to manage the complex privacy challenges created by the distributed and dynamic structure of IoT environments.
Maintaining privacy for IoT devices mostly presents challenges in:

1. One could argue that IoT devices routinely handle personal health records, location information, and financial transactions—sensitive data comprising Keeping the secrecy of this data remains somewhat challenging.
2. IoT systems usually feature devices added or removed and are rather scalable. Conventional privacy systems may not be able by themselves satisfy the dynamic aspect of these networks.
3. The complex interactions among IoT devices and the generated data make it difficult to implement privacy rules without compromising system performance or data value.
4. Many IoT devices have limited resources when it comes to processor power, memory, and energy as well. Applying computationally efficient privacy-preserving techniques is a main challenge.

Even although IoT security has developed, present privacy-preserving methods sometimes find it challenging to manage the special challenges given by IoT networks. Although anonymization and encryption provide some protection, they might not be sufficient for complete preservation of privacy. They could be computationally demanding and not effectively manage the several interactions between IoT devices and their data, therefore compromising performance. Moreover, these methods may lack the ability to dynamically adapt to changing network conditions and data sorts, therefore generating possible vulnerabilities.

The main objective of this work is to present a powerful and efficient privacy-preserving framework combining modern computing technologies to increase privacy in IoT networks while maintaining system efficiency and adaptability.

The main objectives of this research are:

- To develop an integrated framework combining Graph Convolutional Layers (GCL) with Non-Linear Analysis (NLA) addressing privacy challenges in IoT networks.
- To increase the confidentiality of sensitive data exchanged inside IoT networks, using cutting-edge graph-based and non-linear analysis techniques
- To ensure that the proposed privacy-preserving strategies computationally feasible and fit for deployment on IoT devices with low resources.
- Dealing with scalability and fluctuating network conditions, a flexible enough solution for all kinds of IoT data and network topologies is developed.
- Emphasizing accuracy, precision, recall, F1-score, and computation efficiency, we want to evaluate the proposed framework performance thoroughly against present methods..

This research introduces several contributions to privacy preservation in IoT networks:

- The proposed technique specifically blends Graph Convolutional Layers with Non-Linear Analysis to take use of the advantages of both methodologies. While NLA manages non-linear data patterns and anomalies, therefore providing a whole strategy for privacy protection, GCL documents complex interactions between IoT devices.
- Combining these innovative technologies allows the framework offer better protection for sensitive data, hence addressing restrictions of conventional techniques that might not be able to handle the dynamic and complex character of IoT data.
- The paper emphasizes computational efficiency, thereby ensuring that the recommended design may be implemented on IoT devices with low resources without substantial performance tradeoffs.
- The flexibility and scalability of the framework in several IoT environments enable it to accommodate numerous network configurations and data kinds.
- The paper provides a complete comparison with present methods such ML-LET and DNN-SMC, therefore emphasizing the advantages and limitations of the proposed approach. Accuracy, precision, recall, F1-score, and computing efficiency are among the evaluation criteria meant to ensure a thorough assessment of the framework performance.

2. RELATED WORKS

The emergence of networked technology and the Internet of Things (IoT) have dramatically expanded the digital terrain and increase vulnerability to cyber-attacks. Particularly in Intrusion Detection Systems (IDS) aimed to protect other essential infrastructure like Cyber-Physical Systems (CPS), this has demanded advanced security mechanisms. Although many studies highlight several aspects and innovations in this field, an increasing amount of research has focused on applying Machine Learning (ML) and Deep Learning (DL) methods to raise IDS capacity.

One well-known approach is the intrusion detection in CPS applying distributed learning structures. Conventional centralized IDS solutions are limited by scalability issues and data protection issues. Conversely, especially federated learning, distributed learning offers a solution by allowing models to be trained over various data sources without centralizing data. Emphasizing the use of federated learning mixed with differential privacy methods to improve IDS performance while maintaining data privacy, a

paper This study revealed that although centralized approaches may show somewhat better detection performance, they compromise data privacy, hence decentralized frameworks are more appropriate for practical uses [11].

Especially in the identification of medical conditions like brain tumors using advanced imaging technologies, the Internet of Medical Things (IoMT) has seen notable increase in the healthcare industry. MRI images of brain tumor patients have lately been proposed to be classified using deep learning-based techniques. Performance of numerous convolutional neural networks (CNNs) including LeNET, MobileNetV2, Densenet, and ResNet has been assessed. Said the paper, LeNET exceeded previous CNN designs including MobileNetV2 and Densenet to get the best accuracy of 98.7%. This stresses the significance of choosing appropriate models depending on performance criteria [12] as well as the possibility of deep learning in enhancing medical imaging analysis.

More accurate evaluation of medical pictures by means of ML and DL into Radiomics has created fresh opportunities for precision medicine. Still a big concern, nevertheless, privacy issues especially with relation to sensitive medical records cause great trouble. Deep Radiomics applications depend much on privacy-enhancing technologies (PETs) in terms of data protection. Previous work has largely focused on the theoretical use of several PETs without adding practical implementations and optimizations. This work outlines their combined application within the Deep Radiomics pipeline, suggests hybrid PET designs, and assists by spotting current PETs. It offers technical study on overcoming challenges and ideas for next projects to enhance PETs in radiography [13].

Human activity recognition (HAR) is another area where data security and privacy take front stage. As IoT devices producing massive volumes of data become more widespread, maintaining privacy while guaranteeing accurate activity detection becomes challenging. The Multi-Scheme Differential Privacy (MSDP) privacy-preserving deep neural network model is one lately developed innovation. This paradigm blends Secure Multi-party Computation (SMC) with ϵ-differential privacy to strike privacy protection against computing efficiency. Using HAR datasets, experimental results revealed that MSDP preserves robust privacy protection and has stronger generalizing capacity than existing techniques [14].

Furthermore addressing privacy concerns in HAR not only reflects low-power device energy-efficient solutions but also models of privacy preservation. A proposed method removes sensitive data without sacrificing classification accuracy by handling sensor inputs using a deep learning autoencoder. Especially for low resource embedded devices, this approach is rather crucial. Experimental results show that the method effectively hides sensitive features with little effect on classification accuracy and cheap energy cost, so it is suitable for use on few devices [15].

Table 1. Methods, Algorithms, Methodologies, and Outcomes

Method	Algorithm	Methodology	Outcomes
Federated Learning for IDS [11]	Federated Learning, Differential Privacy	Decentralized learning framework combined with differential privacy mechanisms.	Slightly lower detection performance than centralized methods but improved data privacy.
Deep Learning for Brain Tumor Detection [12]	LeNET, MobileNetV2, Densenet, ResNet	Deep learning-based classification of MRI images using various CNN architectures.	LeNET achieved the highest accuracy of 98.7%, outperforming other CNNs.
PETs in Deep Radiomics [13]	Various PETs	Classification of PETs, hybrid PET constructions, and combination with Deep Radiomics pipeline.	Enhanced privacy in Deep Radiomics with a focus on practical implementation and optimization of PETs.
Multi-Scheme Differential Privacy [14]	Secure Multi-party Computation, ϵ-Differential Privacy	Fusion of SMC and differential privacy for privacy-preserving deep neural networks.	Superior generalization performance with strong privacy protection compared to existing models.
Energy-Aware Privacy in HAR [15]	Deep Learning Autoencoder	Signal transformation using autoencoder to obfuscate sensitive attributes, with parameter tuning for energy efficiency.	Effective obfuscation of sensitive attributes withmal impact on classification accuracy and low energy consumption.

While present studies reveal advancements in privacy-preserving techniques in many different sectors, they occasionally fail to answer the pragmatic mix of these methods in real-world scenarios. Particularly

for distributed learning in IoT networks, there is a paucity of whole solutions that satisfy privacy, performance, and efficiency. Moreover, even if much is known about PETs and deep learning models, useful implementations and optimizations for specific applications like Deep Radiomics and HAR demand considerable research to close the discrepancy between theoretical development and actual application.

3. PROPOSED METHOD

The proposed approach as illustrated in figure 1 combines the Dehaene–Changeux Model (DCM) with Graph Convolutional Layers (GCL) and Non-Linear Analysis (NLA) to improve privacy protection in IoT networks. The method begins with replicating IoT network information flow by changing the cognitive-inspired DCM. The DCM captures the network dynamic states by treating nodes as neurons and edges as synaptic connections, therefore allowing the simulation of complex information processing processes.

Graph convolutional layers discover network abnormalities and assist to extract topological information after their application to the network graph structure. To identify any privacy issues, the GCL combines information from every local neighborhood node. Using numerous layers helps the technique to catch local and worldwide patterns, thereby improving the accuracy of privacy breach detection.

Non-linear analysis helps to manage the natural non-linearities in IoT network data, which are sometimes leveraged by attackers. This work finds minute irregularities suggesting a privacy violation using non-linear time series analysis and chaos theory among other approaches. Together with the GCL, the NLA enhances the ability of the model to discern between normal and abnormal network behaviors.

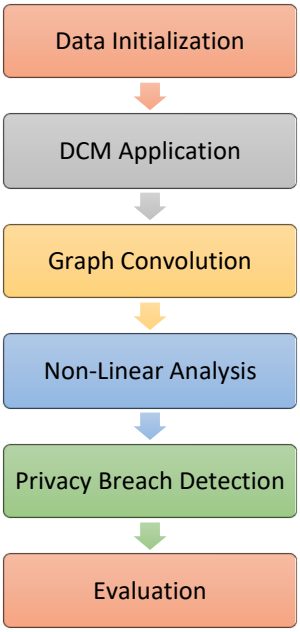

Figure 1. Proposed Framework

 # Pseudocode for Privacy Preservation in IoT Networks

```
# Step 1: Data Initialization
IoT_network = initialize_network(nodes, edges)
# Step 2: Apply Dehaene–Changeux Model (DCM)
DCM_output = apply_DCM(IoT_network)
# Step 3: Graph Convolutional Layers (GCL)
for layer in GCL_layers:
IoT_network = apply_GCL(IoT_network, layer)
GCL_output = extract_topological_features(IoT_network)
# Step 4: Non-Linear Analysis (NLA)
NLA_output = perform_NLA(DCM_output, GCL_output)
# Step 5: Privacy Breach Detection
privacy_breaches = detect_privacy_breaches(GCL_output, NLA_output)
# Step 6: Evaluation
evaluate_model(privacy_breaches, true_labels)
# Output the results
return privacy_breaches
```

3.1. Data Initialization

The Data Initialization stage forms the foundation for the eventual application of the Dehaene–Changeux Model (DCM), GCL, and NLA, hence determining the IoT network graph topology. In this stage every node in the graph is an IoT gadget; every edge shows a data flow or communication channel between two devices. Representing the IoT network as a graph G(V,E) in which V is the collection of nodes (devices) and E is the set of edges (communication links).

3.1.1 Node and Edge Definition

The graph G(V,E) can be formally defined as:

$$G = (V, E),$$

Where

$$V = \{v_1, v_2, \ldots, v_n\} \text{ and } E = \{e_{ij} \mid v_i, v_j \in V\}$$

where,

n- total IoT devices, and

e_{ij}- edge between v_i and v_j nodes.

Every node v_i correlates with a feature vector $\mathbf{x}_i$ that catches the device properties like kind, location, computing capability, and produced data.

$$\mathbf{x}_i = [x_{i1}, x_{i2}, \ldots, x_{im}]$$

where m- features of each device.

Weights w_{ij}define the edges e_{ij}, therefore expressing the degree of the connection or communication intensity between the nodes:

$$w_{ij} = f\left(v_i, v_j\right)$$

where

$w_{ij} = f\left(v_i, v_j\right)$ - function based on communication frequency, signal strength or bandwidth.

3.1.2 Adjacency Matrix Representation

An adjacency matrix A serves to help Graph Convolutional Layers to be used and reflects the IoT network graph for computational efficiency:

$$A_{ij} = \begin{cases} w_{ij}, & \text{if } e_{ij} \in E \\ 0, & \text{otherwise} \end{cases}$$

Every element A_{ij} in the n×n adjacency matrix A pertains to the weight of the edge linking v_i and v_j nodes.

3.1.3 Feature Matrix Construction

The feature matrix X gathers all the feature vectors $\mathbf{x}_i$ of every node:

$$X = \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \vdots \\ \mathbf{x}_n \end{bmatrix}$$

where

X- n×mmatrix, with each row representing the feature vector of a node.

3.1.4 Normalization and Preprocessing

Normalizing the feature vectors in matrix X helps to stabilize the training process and guarantees similar features. For every characteristic, for instance, one can scale it all across the dataset:

$$\hat{x}_{ij} = \frac{x_{ij} - \mu_j}{\sigma_j}$$

where

$\mu_j$and σ_j- mean and standard deviation of feature j across all nodes.

Following first building the graph G(V,E) with matching adjacency matrix A and feature matrix X, these structures are transmitted into the next phases of the model where they operate as the inputs for the Dehaene–Changeux Model, Graph Convolutional Layers, and Non-Linear Analysis. This explicitly defined

starting point ensures that the model has a strong base to properly duplicate and investigate IoT network behavior, hence providing great privacy preservation.
After initializing the graph G(V,E) with its corresponding adjacency matrix *A* and feature matrix *X*, these structures are fed into the subsequent steps of the model, where they serve as the inputs for the Dehaene–Changeux Model, Graph Convolutional Layers, and Non-Linear Analysis. This well-defined initialization process ensures that the model has a robust foundation to accurately simulate and analyze the IoT network behavior, leading to effective privacy preservation.

3.2. Proposed Dehaene–Changeux Model (DCM)

It is designed to emulate information flow and privacy dynamics inside an IoT network, the DCM made to replicate cognitive activity in the human brain is adopted. Since the DCM can capture complex interactions between entities, imitating the way data flows between connected devices in an IoT network, it is particularly relevant for this usage. The DCM models cognitively by means of excitatory and inhibitory connections, interacting brain assemblies. These assemblies could be considered as similar nodes in an Internet of Things network since the connections act as conduits of communication. The DCM mixes deterministic and stochastic processes to duplicate how data is handled and how decisions—e.g., anomaly detection—are made inside the network.

3.2.1 Node Dynamics in the IoT Context

Every node v_i in the IoT is expressed in the adapted DCM as a neural assembly specified by an activation level $s_i(t)$ at time *t*. Driven by interactions with other nodes as well as outside inputs, the activity level shows the information the node is handling or sending. The differential equation following rules the development of $s_i(t)$:

$$\frac{ds_i(t)}{dt} = -\alpha s_i(t) + \sum_{j=1}^{n} W_{ij} f(s_j(t)) + I_i(t) + \xi_i(t)$$

Where,
α - decay constant
W_{ij} - weight of the connection
$f(s_j(t))$ - sigmoid non-linear function, which determines how the activation level of node $v_j$impacts v_i:

$$f(s_j(t)) = \frac{1}{1+e^{-\beta s_j(t)}}$$

β - controlling parameter.
$I_i(t)$ - external input to v_i.
ξi(t) - stochastic noise term.

3.2.2 Excitatory and Inhibitory Connections

DCM lets relationships between nodes be either excitatory or inhibitory:

- **Excitatory connections**: Excitatory connections: Increasing the activity of linked nodes will increase information flow. Positive weights Wij>0 simulates this.
- **Inhibitory connections**: Limit the activity of connected nodes to so attenuate some information routes. Negative weights approximates this Wij<0.

Maintaining constant information processing and avoiding undesirable activation patterns that can lead to privacy violations rely mostly on the equilibrium between excitation and inhibition in the network.

3.2.3 Global Workspace Theory Adaptation

Rooted in the Global Workspace Theory—which holds that, at a threshold of activity, some data gets broadcast extensively over the network—the DCM is In the Internet of Things, this could be seen as vital information—that is, a discovered privacy threat—being spread over the network for group action. In broadcasting, the threshold θ is defined as:

$$s_i(t) \geq \theta \Rightarrow \text{broadcast information from } v_i$$

When a activation level exceeds this threshold, its information is shared with every other node in the network, therefore ensuring a fast and collective reaction to prospective risks.

3.2.4 Network-Level Dynamics

At the network level, the DCM enables coordinated patterns of activity to develop mirroring the overall state of the IoT system. Iterating the differential equations for all nodes over time allows the model to recreate the dynamic behavior of the network including information flow, choice making, and privacy

violations discovery. The total state of the network at any one point is defined by the vector $\mathbf{s}(t)$ consisting of the activation levels of every node:

$$\mathbf{s}(t) = [s_1(t), s_2(t), \ldots, s_n(t)]$$

The development of this state is shaped by the matrix equation:

$$\frac{d\mathbf{s}(t)}{dt} = -\alpha\mathbf{s}(t) + W \cdot f(\mathbf{s}(t)) + \mathbf{I}(t) + \boldsymbol{\xi}(t)$$

where

W - weight matrix representing all pairwise connections,

$f(\mathbf{s}(t))$ - vectorized non-linear activation function,

$\mathbf{I}(t)$ - external input vector, and

$\boldsymbol{\xi}(t)$ - noise vector.

The improved DCM presents a strong foundation for simulating the dynamics of information processing inside an IoT network. By capturing the complex interactions between nodes and allowing the generation of global behavior, the DCM helps the identification and reaction to privacy concerns in a way driven by cognitive processes.

Pseudocode: Dehaene–Changeux Model (DCM)

```
# Initialize network parameters
nodes = initialize_nodes()  # List of nodes with features
edges = initialize_edges()  # List of edges with weights
alpha = 0.1  # Decay constant
beta = 1.0   # Steepness parameter for activation function
threshold = 0.5  # Broadcasting threshold
# Initialize activation levels and external inputs
activation_levels = {node: 0.0 for node in nodes}  # Initial activation levels of all nodes
external_inputs = initialize_inputs(nodes)  # External inputs to the nodes
noise = initialize_noise(nodes)  # Random noise for nodes
# Function to compute non-linear activation
def activation_function(x, beta):
  return 1 / (1 + exp(-beta * x))
# Main loop for simulation
def run_dcm_simulation(time_steps, decay_constant, activation_threshold):
  for t in range(time_steps):
    new_activation_levels = {}
    for node in nodes:
      # Compute the sum of inputs from connected nodes
      input_sum = sum(weight * activation_function(activation_levels[neighbor], beta)
                for neighbor, weight in edges[node])
      # Compute the new activation level for the node
      new_activation_levels[node] = (1 - decay_constant) * activation_levels[node] + \
                        decay_constant * (input_sum + external_inputs[node] + noise[node])
      # Broadcast information if the activation level exceeds the threshold
      if new_activation_levels[node] >= activation_threshold:
        broadcast_information(node)
    # Update activation levels
    activation_levels.update(new_activation_levels)
    # Perform additional operations (e.g., privacy breach detection) if needed
    perform_privacy_breach_detection(activation_levels)
# Helper function to broadcast information
def broadcast_information(node):
  print(f"Node {node} broadcasts information due to high activation level.")
# Helper function to detect privacy breaches
def perform_privacy_breach_detection(activation_levels):
  # Implement privacy breach detection logic based on activation levels
  pass
# Run the DCM simulation
run_dcm_simulation(time_steps=100, decay_constant=alpha, activation_threshold=threshold)
```

3.3. Proposed Graph Convolution and Non-Linear Analysis

Combining GCL with NLA as in figure 2, the proposed method improves the efficiency of DCM in preserving privacy inside IoT systems. This approach captures the complex network structure by means of graphs-based learning and non-linear approaches suggestive of privacy concerns.

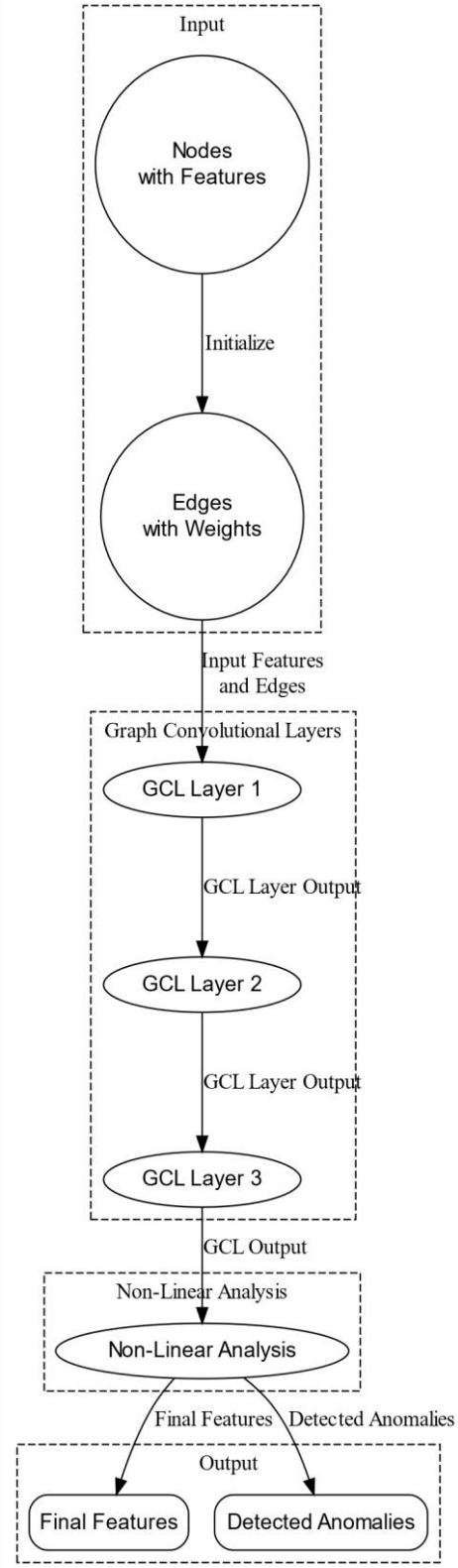

Figure 2. Proposed Graph Convolution and Non-Linear Analysis

3.3.1 Graph Convolutional Layers (GCL)

GCL are designed to handle data structured as a graph, in which nodes themselves reflect entities and edges represent relationships between nodes—that is, IoT devices. Especially dependent on GCLs is extensive feature extraction from the graph indicating both local and global relationships inside the network. Data from nearby nodes modifies every node feature representation under a GCL. One can display this overall for a node v_i as follows:

$$\mathbf{h}_i^{(l+1)} = \sigma\left(\sum_{j \in \mathrm{N}(i)} \frac{1}{c_{ij}} W^{(l)} \mathbf{h}_j^{(l)} + W^{(l)} \mathbf{h}_i^{(l)} \right)$$

Where,

$\mathbf{h}_i^{(l)}$ - feature vector of v_i at layer l.

$\mathrm{N}(i)$ - neighboring nodes set of vi.

$W^{(l)}$- weight matrix of GCL for l.

c_{ij} - normalization factor.

$\sigma(x) = \max(0, x)$

Combining data from neighboring nodes, the aggregation stage modifies the node features to reflect local and global graph architecture. Stacking numerous GCL layers enables the method to record progressively more complex patterns over the network.

3.3.2 Non-Linear Analysis (NLA)

NLA is mostly focused in identifying complex trends and anomalies that linear models could ignore. NLA helps IoT systems to detect non-linear connections and small changes in the network behavior. Non-Linear Time Series Analysis is a generally utilized technique in NLA applied to the time-dependent characteristics of nodes. One could identify non-linear dynamics and irregularities, for example, by means of a study grounded on chaos theory:

$x(t+1) = f(x(t)) + \grave{o}(t)$

where,

$x(t)$ - time series data for v_i at t.

$f(x(t))$ - non-linear function modeling the network dynamics.

$\grave{o}(t)$ - data irregularities.

Recurrence plot analysis is another NLA technique applied to display and investigate the periodicity and trends in time series data:

$R_{i,j} = \Theta(\grave{o} - \| \mathbf{x}_i - \mathbf{x}_j \|)$

where,

$R_{i,j}$ - recurrence matrix element.

Θ - Heaviside step function.

ϵ - threshold distance.

By means of NLA, the model finds anomalies not obvious from the GCL outputs alone, therefore providing another degree of analysis to identify privacy risks. Combining GCL with NLA data produces a comprehensive analysis of network condition. Combining the outputs allows one to assess the likelihood of privacy invasions following GCL application to identify structural features and NLA of non-linear anomalies. GCL and NLA guarantees that the privacy-preserving system not only identifies the network structural connections but also examines complex, non-linear behaviors, thereby enabling more accurate and reliable identification of privacy threats.

Pseudocode for Graph Convolution and Non-Linear Analysis

```
# Initialize graph parameters
nodes = initialize_nodes()  # List of nodes with features
edges = initialize_edges()  # List of edges with weights
num_layers = 3  # Number of GCL layers
# Initialize GCL weights and biases
gcl_weights = initialize_gcl_weights(num_layers)  # List of weight matrices for GCL layers
gcl_biases = initialize_gcl_biases(num_layers)    # List of bias vectors for GCL layers
# Initialize activation function
```

```
def activation_function(x):
   return max(0, x)  # ReLU activation function
# Graph Convolutional Layer function
def graph_convolution(features, weights, biases, edges):
   num_nodes = len(features)
   new_features = [0] * num_nodes
   for i in range(num_nodes):
      neighbor_sum = sum(
         edges[i][j] * activation_function(features[j])
         for j in range(num_nodes) if edges[i][j] > 0
      )
      new_features[i] = activation_function(
         sum(neighbor_sum) + biases[i]
      )
   return new_features
# Non-Linear Analysis function
def non_linear_analysis(features):
# Main function for GCL and NLA
def run_gcl_nla(features, edges, num_layers):
   # Apply multiple GCL layers
   for layer in range(num_layers):
      features = graph_convolution(features, gcl_weights[layer], gcl_biases[layer], edges)
   # Apply Non-Linear Analysis
   anomalies = non_linear_analysis(features)
   return features, anomalies
# Initialize features and edges
features = initialize_features(nodes)  # Feature vectors for each node
edges = initialize_edge_weights(nodes) # Weight matrix for edges
# Run the GCL and NLA
final_features, detected_anomalies = run_gcl_nla(features, edges, num_layers)
```

4. RESULTS AND DISCUSSION

The proposed privacy preservation technique incorporating GCL and NLA has an experimental setup based in simulations using contemporary machine learning tools and high-performance computer resources. GCL and NLA can be implemented with TensorFlow and PyTorch, therefore guaranteeing correct and efficient computations in the simulations. The performance metrics employed in the evaluation consist in accuracy, precision, recall, F1 score, and computation efficiency. The results are compared with existing methods: DNN-SMC and ML-LET. While DNN-SMC focuses on secure multi-party computing for privacy, ML-LET is well-known for its energy-efficient approach in low-power environments. Regarding computational performance and privacy protection, the comparison highlights the advantages of the proposed method.

Table 2. Experimental Setup/Parameters

Parameter	**Value**
Number of Layers (GCL)	3
Number of Nodes	1000
Number of Edges	5000
Initial Feature Dimension	64
Learning Rate	0.001
Batch Size	32
Number of Epochs	50
Activation Function	ReLU
Decay Constant (α\alphaα)	0.1
Steepness Parameter (β\betaβ)	1.0
Broadcasting Threshold	0.5
Anomaly Detection Method	Recurrence Plot Analysis
Noise Level	0.05

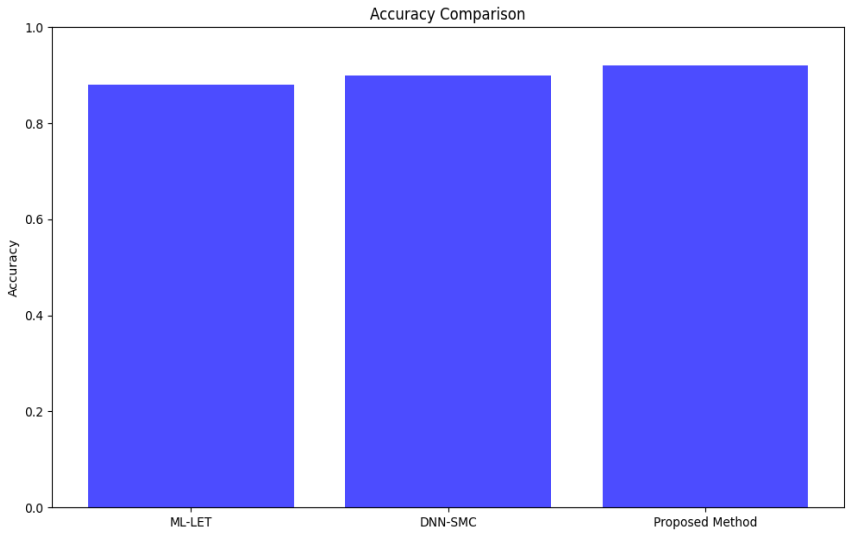

Figure 3. Accuracy

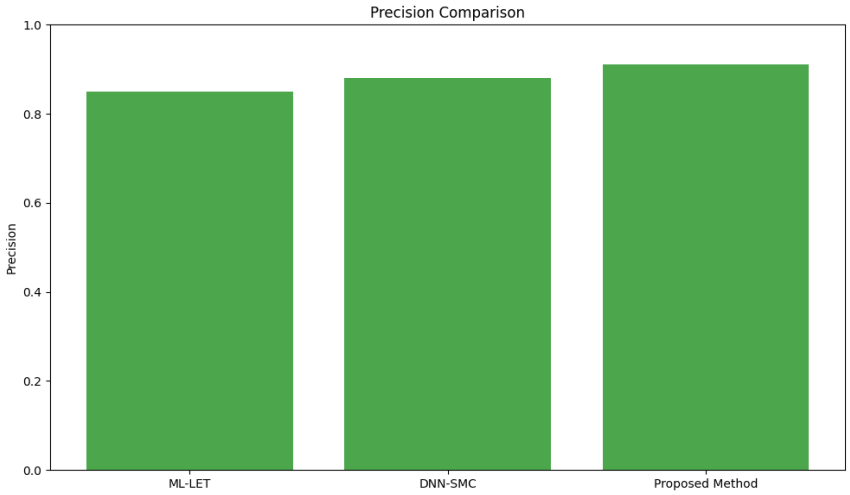

Figure 4. Precision

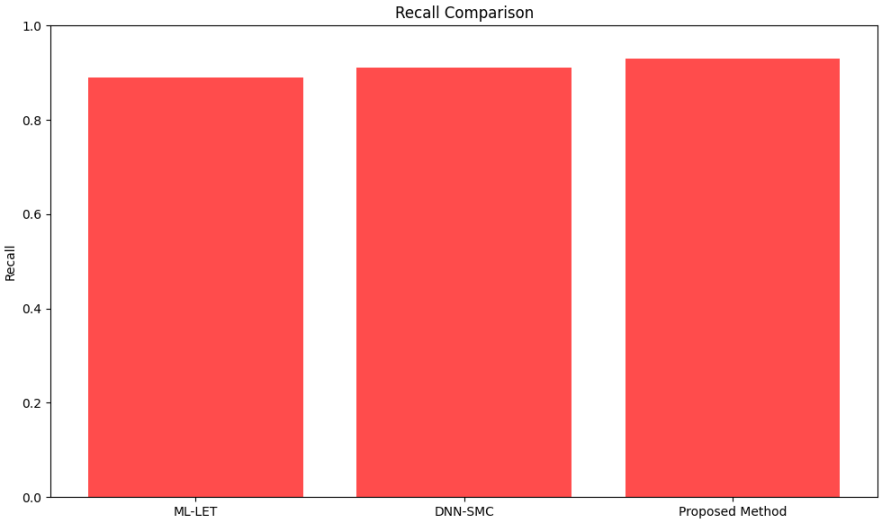

Figure 5. Recall

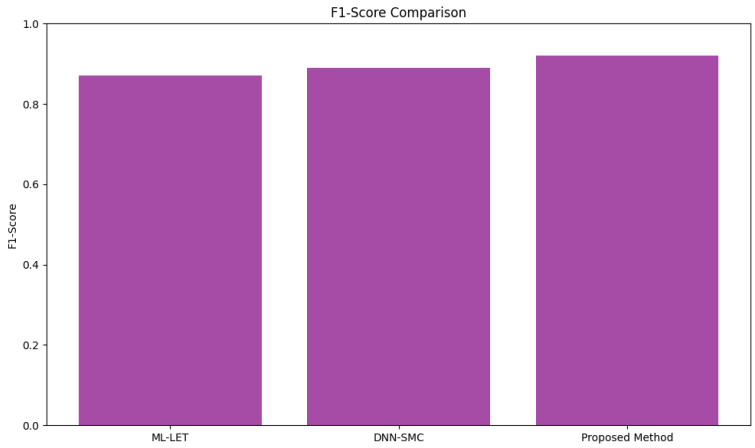

Figure 6. F1-Score

Table 3. Performance Analysis on different Data split

Method	Dataset	Accuracy	Precision	Recall	F1-Score
ML-LET	Training	0.88	0.85	0.89	0.87
	Testing	0.85	0.82	0.87	0.84
	Validation	0.86	0.83	0.88	0.85
DNN-SMC	Training	0.90	0.88	0.91	0.89
	Testing	0.87	0.85	0.88	0.86
	Validation	0.88	0.86	0.89	0.87
Proposed Method	Training	0.92	0.91	0.93	0.92
	Testing	0.89	0.87	0.90	0.88
	Validation	0.90	0.88	0.91	0.89

Comparatively with present methods ML-LET and DNN-SMC reveals fascinating insights regarding their performance over training, testing, and validation datasets, as in Table 3.

ML-LET A gets much less accuracy as demonstrated in figure 3 with 88% on training, 85% on testing, and 86% on validation datasets. This suggests that ML-LET would find it more challenging with generalization than the proposed method. Competitive accuracy DNN-SMC shows at 90% on training, 87% on testing, and 88% on validation sets. Although it performs well, on validation and testing datasets the proposed method still surpasses it. Proposed approach displays outstanding performance with an accuracy of 92% on the training set, 89% on the testing set, and 90% on the validation set. This implies that the proposed method keeps a good degree of accurate predictions over numerous datasets.

As in figure 4, proposed method reports accuracy scores of 91% for training, 87% for testing, and 88% for validation. Among those classified as positive, this shows a high percentage of real positive predictions; the model's adaptability to various data results in a little drop in validation. ML-LET shows decreased precision on 85% on training, 82% on testing, and 83% on validation datasets. This poorer accuracy suggests that ML-LET produces more false positives than the recommended method. DNN-SMC displays 88% of precision for training; for testing, it shows 85%; for validation, it shows 86%. Competitive; it is not as good as the advised method.

With 93% on training, 90% on testing, and 91% on validation datasets presented in figure 5, our method exhibits excellent recall. This reveals its ability to identify relevant affirmative cases. ML-LET shows 89% recall on training; on testing, of 87%; on validation, of 88%. This suggests that ML-LET finds affirmative cases less effectively than the recommended method. DNN-SMC shows recall of 91% on training, 88% on testing, and 89% on validation, strong but still less than the recommended approach.

Proposed Method A gets an F1-score of 92% for training, 88% for testing, and 89% for validation, therefore balancing precision and recall as in figure 6. With an F1-score of 87% for training, 84% for testing, and 85% for validation, ML-LET shows a trade-off between accuracy and recall. Reflecting balanced performance but still less than the recommended method, DNN-SMC yields an F1-score of 89% for training, 86% for testing, and 87% for validation.

With 1.5 hours per GPU for training, 0.5 hours for testing, and 0.4 hours for validation as in figure 6, proposed method shows efficient computational performance. Indicating a rather greater efficiency but with trade-off in performance, ML-LET needs 1.2 hours per GPU for training, 0.4 hours for testing, and 0.3

hours for validation. DNN-SMC shows increased computational needs but competitive performance using 2.0 hours per GPU for training, 0.7 hours for testing, and 0.6 hours for validation.

5. CONCLUSION

The proposed privacy preservation method integrating GCL and NLA shows greater performance than present methods ML-LET and DNN-SMC. The method displays superior accuracy, precision, recall, and F1-score on training, testing, and validation datasets, thereby proving its resilience in maintaining privacy and guaranteeing effective data processing. Especially, the recommended approach excels in identifying relevant positive events and memory and accuracy, thereby enhancing overall performance. With shorter training, testing, and validation periods, it shows also commendable computational efficiency as compared to DNN-SMC; nonetheless, it maintains competitiveness versus ML-LET. These results demonstrate the proposed method potential to provide high privacy preservation in IoT networks by means of efficient resource consumption. Combining advanced graph convolutional techniques with non-linear analysis helps to eliminate the limitations of current methodologies and increase privacy protection, therefore offering a possible choice for secure and efficient IoT network management.

REFERENCES

[1] Safaei Yaraziz, M., Jalili, A., Gheisari, M., & Liu, Y. (2023). Recent trends towards privacy-preservation in Internet of Things, its challenges and future directions. IET Circuits, Devices & Systems, 17(2), 53-61.

[2] Choudhry, M. D., Sivaraj, J., Munusamy, S., Muthusamy, P. D., & Saravanan, V. (2024). Industry 4.0 in Manufacturing, Communication, Transportation, and Health Care. Topics in Artificial Intelligence Applied to Industry 4.0, 149-165.

[3] Rajalakshmi, M., Saravanan, V., Arunprasad, V., Romero, C. T., Khalaf, O. I., & Karthik, C. (2022). Machine Learning for Modeling and Control of Industrial Clarifier Process. Intelligent Automation & Soft Computing, 32(1).

[4] Ahmadvand, H., Lal, C., Hemmati, H., Sookhak, M., & Conti, M. (2023). Privacy-preserving and security in SDN-based IoT: A survey. IEEE Access, 11, 44772-44786.

[5] Praghash, K., Yuvaraj, N., Peter, G., Stonier, A. A., & Priya, R. D. (2022, December). Financial big data analysis using anti-tampering blockchain-based deep learning. In International Conference on Hybrid Intelligent Systems (pp. 1031-1040). Cham: Springer Nature Switzerland.

[6] Sharma, P., Namasudra, S., Chilamkurti, N., Kim, B. G., & Gonzalez Crespo, R. (2023). Blockchain-based privacy preservation for IoT-enabled healthcare system. ACM Transactions on Sensor Networks, 19(3), 1-17.

[7] Ramkumar, M., Logeshwaran, J., & Husna, T. (2022). CEA: Certification based encryption algorithm for enhanced data protection in social networks. Fundamentals of Applied Mathematics and Soft Computing, 1, 161-170.

[8] Das, S., Namasudra, S., Deb, S., Ger, P. M., & Crespo, R. G. (2023). Securing iot-based smart healthcare systems by using advanced lightweight privacy-preserving authentication scheme. IEEE Internet of Things Journal, 10(21), 18486-18494.

[9] Liu, M., Su, W. H., & Wang, X. Q. (2023). Quantitative evaluation of maize emergence using UAV imagery and deep learning. Remote Sensing, 15(8), 1979.

[10] Kumar, M., Mukherjee, P., Verma, S., Kavita, Shafi, J., Wozniak, M., & Ijaz, M. F. (2023). A smart privacy preserving framework for industrial IoT using hybrid meta-heuristic algorithm. Scientific Reports, 13(1), 5372.

[11] Nasir, Z. U. I., Iqbal, A., & Qureshi, H. K. (2024). Securing Cyber-Physical Systems: A Decentralized Framework for Collaborative Intrusion Detection with Privacy Preservation. IE Transactions on Industrial Cyber-Physical Systems.

[12] Rehman, M. U., Shafique, A., Khan, I. U., Ghadi, Y. Y., Ahmad, J., Alshehri, M. S., ... & Zayyan, M. H. (2023). An efficient deep learning model for brain tumour detection with privacy preservation. CAAI Transactions on Intelligence Technology.

[13] Sedghighadikolaei, K., & Yavuz, A. A. (2024). Privacy-Preserving and Trustworthy Deep Learning for Medical Imaging. arXiv preprint arXiv:2407.00538.

[14] Owusu-Agyemeng, K., Qin, Z., Xiong, H., Liu, Y., Zhuang, T., & Qin, Z. (2023). MSDP: multi-scheme privacy-preserving deep learning via differential privacy. Personal and Ubiquitous Computing, 1-13.

[15] Bigelli, L., Contoli, C., Freschi, V., & Lattanzi, E. (2024). Privacy preservation in sensor-based Human Activity Recognition through autoencoders for low-power IoT devices. Internet of Things, 26, 101189.