Graph Neural Network-Based Trust Modeling and Nature-Inspired Optimization for Secure Decision Support Systems

Rajesh Verma[1*], **Dr. Ravindra Kumar Tiwari**[1], **Dr. Jitendra Kumar Agrawal**[2]

[1*]Department of Computer Science & Engineering, LNCT University, JK Town, Kolar Road, Bhopal - 462024, Madhya Pradesh, India | Email: rv097227@gmail.com

[1]LNCT University, JK Town, Kolar Road, Bhopal - 462024, Madhya Pradesh, India | Email: ravindratiwari.s@gmail.com

[2]Lakshmi Narain College of Technology (MCA), LNCT Campus, Kalchuri Nagar, Raisen Road, P.O. Kolua, Bhopal - 462022, Madhya Pradesh, India | Email: agrawaljitendra22@gmail.com

***Corresponding Author:** Rajesh Verma, Department of Computer Science & Engineering, LNCT University, JK Town, Kolar Road, Bhopal - 462024, Madhya Pradesh, India | Email: rv097227@gmail.com

Abstract–We present the Nature-Inspired Graph Neural Network Trust Routing Engine (NIGNN-TRE), a novel subsystem that reformulates routing and data aggregation for Decision Support Systems (DSS). The proposed method addresses the critical challenge of secure data aggregation in dynamic, potentially adversarial network environments. Our architecture integrates four novel components: a Spatiotemporal Graph Constructor that builds dynamic graphs from network telemetry, a Trust-Aware Graph Convolutional Network (GCN) Encoder that computes per-node trustworthiness scores, a Pheromone-Mapped Ant Colony Optimization (ACO) Router that modulates routing decisions using these scores, and a Flocking Consensus Validator that detects colluding nodes through a bio-inspired consensus mechanism. The GCN encoder employs Chebyshev spectral graph convolutions and temporal convolutional layers to learn spatiotemporal patterns of benign and malicious behavior from historical data. The ACO router then replaces static forwarding tables with a probabilistic selection process where pheromone levels are augmented by trust scores, thereby dynamically avoiding untrustworthy nodes. Furthermore, the flocking validator applies a discrete adaptation of the Couzin model to compute consensus states from reported sensor readings; nodes whose deviations exceed an adaptive threshold are suppressed from future routing decisions. We integrate this subsystem into the existing DSS communication module, where it interfaces directly with the message queue, network interface, and decision engine. The key contributions include a hybrid nature-inspired architecture that fuses graph neural networks with swarm intelligence, a trust-aware pheromone update rule that incorporates learned trust scores, and a flocking-based consensus mechanism that provides resilience against collusion attacks. This work consequently enables secure, adaptive, and trustworthy data aggregation without requiring modifications to the underlying security layer.

Keywords: Graph Neural Networks, Trust Modeling, Nature-Inspired Optimization, Ant Colony Optimization, Decision Support Systems, Secure Data Aggregation, Network Security.

How to cite this article: Verma R, Tiwari RK, Agrawal JK. Graph Neural Network-Based Trust Modeling and Nature-Inspired Optimization for Secure Decision Support Systems. Int J Drug Deliv Technol. 2026;16(70s): 187-201. DOI: 10.25258/ijddt.16.70s.18

I. INTRODUCTION

Decision Support Systems (DSS) have become indispensable tools for aiding complex decision-making processes across diverse domains, including healthcare, finance, and industrial control [1]. These systems rely on the timely and accurate aggregation of data from distributed sensor networks to generate actionable insights [2]. However, the increasing deployment of DSS in dynamic and potentially adversarial environments introduces significant security challenges, particularly concerning the integrity of routing and data aggregation subsystems [3]. Malicious actors can compromise individual sensor nodes to inject false data, manipulate aggregated metrics, or launch collusion attacks that subvert the entire decision-making pipeline [4].

Traditional approaches to securing data aggregation in DSS have predominantly relied on cryptographic protocols and static trust management frameworks [5]. While these methods provide a baseline level of security, they often exhibit limited adaptability against sophisticated, evolving threats such as coordinated collusion attacks [6]. Cryptographic solutions, for instance, cannot detect nodes that are authenticated but behave

maliciously, while static trust models fail to capture the dynamic nature of node reliability in changing network topologies [7]. Consequently, there is a pressing need for adaptive, intelligent security mechanisms that can continuously learn from network behavior and autonomously respond to emerging threats.

Nature-inspired computing has emerged as a powerful paradigm for addressing complex optimization and security problems in distributed systems [8]. Algorithms such as Ant Colony Optimization (ACO) and Particle Swarm Optimization (PSO) have been successfully applied to routing problems in wireless sensor networks, demonstrating remarkable ability to find optimal paths while avoiding congested or compromised nodes [9], [10]. These bio-inspired approaches mimic the collective intelligence observed in natural systems, enabling decentralized decision-making and self-adaptation without requiring centralized control [11]. However, existing nature-inspired routing protocols typically rely on simple heuristics or local information, which may be insufficient to detect subtle patterns of malicious behavior in complex network environments.

In parallel, recent advances in deep learning, particularly Graph Neural Networks (GNNs), have revolutionized the analysis of non-Euclidean data structures such as network topologies [12]. Spatiotemporal Graph Convolutional Networks (ST-GCNs) extend this capability by capturing both spatial dependencies and temporal dynamics, making them ideally suited for modeling the evolving behavior of sensor nodes over time [13]. These models can learn complex patterns of benign and malicious activity from historical network telemetry, providing a robust foundation for trust assessment. Nevertheless, the integration of GNN-based trust reasoning with nature-inspired routing mechanisms remains largely unexplored in the context of DSS security.

This paper proposes the Nature-Inspired Graph Neural Network Trust Routing Engine (NIGNN-TRE), a novel subsystem that synergistically combines spatiotemporal graph convolutional networks with ant colony optimization and flocking-based consensus to secure routing and data aggregation in DSS. The proposed method employs a temporal graph neural network to continuously analyze dynamic network topology and historical reporting latency, thereby computing real-time trustworthiness scores for each sensor node. These quantified trust metrics are subsequently mapped onto virtual pheromone trails within a decentralized ACO routine, guiding data aggregation paths toward high-reliability routes while naturally evaporating low-trust pathways. Furthermore, a flocking-based consensus mechanism, inspired by the Couzin model of collective animal behavior, is applied to detect and isolate colluding nodes attempting to manipulate aggregated metrics [14]. This synergistic integration of graph-based anomaly reasoning and nature-inspired swarm routing ensures resilient, tamper-resistant data transmission across the distributed decision support environment.

The key contributions of this work are threefold. First, we present a hybrid architecture that fuses graph neural networks with swarm intelligence, enabling trust-aware routing decisions that adapt to dynamic network conditions. Second, we introduce a trust-aware pheromone update rule that incorporates learned trust scores from the GCN encoder, thereby replacing static forwarding tables with a probabilistic selection process that dynamically avoids untrustworthy nodes. Third, we propose a flocking-based consensus mechanism that provides resilience against collusion attacks by computing consensus states from reported sensor readings and suppressing nodes whose deviations exceed an adaptive threshold. Unlike existing approaches that treat trust assessment and routing as separate concerns, our framework integrates these functions into a cohesive, self-adaptive subsystem that operates without requiring modifications to the underlying security layer.

The remainder of this paper is organized as follows. Section 2 reviews related work on nature-inspired routing, graph neural networks for network security, and trust management in DSS. Section 3 presents the necessary preliminaries, including background on spatiotemporal graph convolutional networks, ant colony optimization, and flocking consensus models. Section 4 details the proposed NIGNN-TRE architecture, describing the spatiotemporal graph constructor, trust-aware GCN encoder, pheromone-mapped ACO router, and flocking consensus validator. Section 5 presents experimental evaluation results, demonstrating the effectiveness of our approach against various attack scenarios. Section 6 discusses the implications of our findings and outlines directions for future research. Section 7 concludes the paper.

II. Related Work

The problem of secure data aggregation in decision support systems intersects several active research domains, including nature-inspired routing protocols, graph neural network-based anomaly detection, and trust management frameworks. This section reviews the most relevant contributions in each area, highlighting their strengths and limitations in the context of our proposed NIGNN-TRE architecture.

A. Nature-Inspired Routing and Swarm Intelligence

Nature-inspired algorithms have been extensively applied to routing problems in wireless sensor networks and IoT environments. Ant Colony Optimization (ACO), in particular, has demonstrated remarkable

effectiveness in discovering optimal paths through decentralized, probabilistic decision-making [9]. Early works such as AntNet and AntHocNet applied ACO to packet routing in wired and mobile ad hoc networks, respectively, using virtual pheromone trails to encode path quality metrics like latency and hop count [15]. More recent studies have extended these ideas to secure routing, where pheromone levels are modulated by trust scores or reputation values to avoid compromised nodes [16]. For example, the Trust-Aware Ant Colony Routing (TA-ACR) protocol incorporates direct and indirect trust observations into the pheromone update rule, enabling the network to dynamically bypass malicious nodes [17].

However, these trust-aware ACO approaches typically rely on simple, locally computed trust metrics, such as packet forwarding ratio or energy consumption, which may be insufficient to detect sophisticated attacks like collusion or data injection [18]. The trust assessment in these methods is often static or updated at coarse time intervals, limiting their adaptability to rapidly changing network conditions. Furthermore, the heuristic desirability term in the probabilistic path selection formula is usually based on physical distance or hop count, which does not capture the complex spatiotemporal patterns of malicious behavior. Our proposed PM-ACO component addresses these limitations by replacing the static trust metric with a learned trust score from a spatiotemporal graph neural network, thereby enabling more nuanced and adaptive routing decisions.

Particle Swarm Optimization (PSO) has also been applied to routing and clustering problems in sensor networks, where particles represent potential routes or cluster heads and the swarm converges toward optimal solutions through social learning [10]. While PSO-based approaches can achieve good convergence properties, they often require centralized coordination or global knowledge of the network state, which may be impractical in large-scale, dynamic DSS environments [19]. In contrast, ACO's decentralized, stigmergic communication mechanism is better suited for distributed decision-making, as each ant only requires local information to update pheromone trails.

B. Graph Neural Networks for Network Security

Graph Neural Networks (GNNs) have emerged as a powerful tool for analyzing network-structured data, with applications ranging from social network analysis to intrusion detection [12]. In the context of network security, GNNs can model the topological relationships between nodes (e.g., sensor devices, routers) and learn patterns of normal and anomalous behavior from historical data [20]. Spatiotemporal Graph Convolutional Networks (ST-GCNs) extend this capability by incorporating temporal dynamics, enabling the detection of time-evolving attack patterns such as gradual data injection or slow-rate denial-of-service attacks [13].

Several recent works have applied GNNs to trust management and anomaly detection in IoT and sensor networks. The TrustAware-GNN framework, for instance, uses a graph attention network to compute trust scores for IoT devices based on their interaction history and network topology [21]. This approach achieves real-time trust inference by aggregating information from neighboring nodes through attention mechanisms, thereby capturing both direct and indirect trust evidence. Similarly, DTrust employs a dynamic graph neural network to assess trust levels in time-varying online social networks, using a predefined aggregation method to establish trust inference models [22]. These methods demonstrate the effectiveness of GNNs for trust assessment, but they typically focus on static or slowly evolving graphs and do not explicitly address the routing problem.

In the domain of intrusion detection, ST-GCNs have been applied to detect zero-day attacks in UAV swarms by modeling the spatiotemporal dynamics of network traffic [23]. The Defense-Aware Temporal Graph Neural Network (DA-TGNN) extends this idea to IIoT-5G environments, incorporating defense mechanisms that adapt to evolving attack strategies [24]. However, these works primarily focus on detection rather than mitigation; they output anomaly scores or alerts but do not directly influence routing or data aggregation decisions. Our TA-GCN component bridges this gap by outputting trust scores that are directly consumed by the ACO router and flocking validator, thereby enabling proactive avoidance of untrustworthy nodes.

C. Trust Management and Consensus in Decision Support Systems

Trust management is a critical component of secure decision support systems, as it determines which data sources can be relied upon for generating accurate recommendations [5]. Traditional trust management frameworks in DSS often employ centralized reputation systems, where a trusted authority collects and aggregates trust reports from all nodes [25]. While these systems can provide robust trust assessment, they introduce a single point of failure and may not scale well to large, dynamic networks [26]. Decentralized trust management approaches, such as those based on blockchain or distributed consensus, address these limitations by distributing trust assessment across multiple nodes [27].

Consensus mechanisms, originally developed for distributed ledger technologies, have been adapted for data aggregation in sensor networks to ensure the integrity of aggregated metrics [28]. The flocking-based consensus model, inspired by the collective behavior of

birds and fish, offers a particularly attractive approach for distributed decision-making in dynamic environments [14]. In this model, each node updates its state based on the average of its neighbors' states, with a coupling term that pulls the node toward its own sensor reading. This mechanism naturally filters out outliers, as nodes with anomalous readings will deviate from the consensus state and can be detected. Our FCV component extends this idea by incorporating trust scores into the coupling gain, thereby giving more weight to high-trust nodes and suppressing the influence of potentially malicious nodes.

D. Comparison with Existing Works

While the individual components of our proposed NIGNN-TRE—nature-inspired routing, GNN-based trust assessment, and flocking consensus—have been explored in isolation, their synergistic integration into a unified subsystem for secure data aggregation in DSS represents a novel contribution. Existing trust-aware routing protocols typically rely on simple, locally computed trust metrics that cannot capture complex spatiotemporal attack patterns. Conversely, GNN-based anomaly detection methods often stop at generating alerts without directly influencing routing decisions. Furthermore, consensus mechanisms for data aggregation are usually decoupled from the routing layer, leading to potential inconsistencies between the data paths and the validation logic. Our framework addresses these gaps by establishing a closed-loop system where the GNN encoder continuously learns trust scores from network telemetry, the ACO router uses these scores to guide routing decisions, and the flocking validator provides a second line of defense against collusion attacks. This integrated approach ensures that only high-trust nodes participate in data aggregation, thereby enhancing the resilience and reliability of the decision support system.

III. PRELIMINARIES

To ensure a self-contained exposition, this section introduces the foundational concepts that underpin the proposed NIGNN-TRE architecture. We first formalize the data aggregation problem in a decision support system as a graph-based optimization task. Subsequently, we provide essential background on spatiotemporal graph convolutional networks for trust inference, ant colony optimization for probabilistic routing, and flocking-based consensus models for outlier detection.

A. Problem Formulation

We model the sensor network underlying a DSS as a dynamic, directed graph $\mathcal{G}_t = (\mathcal{V}, \mathcal{E}_t)$, where $\mathcal{V} = \{v_1, v_2, \dots, v_N\}$ is the set of N sensor nodes, and $\mathcal{E}_t \subseteq \mathcal{V} \times \mathcal{V}$ is the set of communication links present at time step t. Each node v_i periodically generates a sensor reading $s_i(t) \in \mathbb{R}^d$ and forwards it toward a sink node v_0 (the decision engine) via a multi-hop path. The data aggregation subsystem must compute an aggregated metric $M(t) = f(\{s_i(t): v_i \in \mathcal{V}\})$ at the sink, where f is an aggregation function (e.g., mean, median, or sum). The objective is to compute $M(t)$ reliably even when a subset of nodes $\mathcal{A} \subset \mathcal{V}$ is compromised and injects false readings $\tilde{s}_i(t) \neq s_i(t)$.

This problem can be cast as a constrained optimization over a time-varying graph. At each time step t, we must select a set of source nodes $\mathcal{S}(t) \subseteq \mathcal{V}$ and a routing tree $\mathcal{T}(t)$ rooted at the sink such that the expected error $\| M(t) - f(\{s_i(t): v_i \in \mathcal{S}(t)\}) \|$ is minimized, subject to constraints on communication latency, energy consumption, and the unknown trustworthiness of individual nodes. The dynamic nature of the graph—where edges may appear or disappear due to mobility, interference, or node failures—and the adversarial behavior of compromised nodes make this a challenging problem.

B. Spatiotemporal Graph Convolutional Networks

Graph Neural Networks (GNNs) generalize convolutional neural networks to non-Euclidean domains by learning localized filters on graph-structured data [12]. Among the various GNN architectures, Chebyshev spectral graph convolutional networks (ChebNets) offer a computationally efficient approximation of spectral convolutions by using Chebyshev polynomials of the graph Laplacian [29]. The graph signal $\mathbf{x}^{(l)} \in \mathbb{R}^{N \times d_l}$ at layer l is transformed as:

$$\mathbf{x}^{(l+1)} = \sigma\left(\sum_{k=0}^{K} \theta_k \, T_k(\tilde{\mathbf{L}})\mathbf{x}^{(l)}\mathbf{W}^{(l)}\right) \quad (1)$$

where $\tilde{\mathbf{L}}$ is the scaled graph Laplacian, T_k is the Chebyshev polynomial of order k, θ_k are learnable coefficients, $\mathbf{W}^{(l)}$ is a weight matrix, and σ is an activation function (e.g., ReLU).

To capture temporal dependencies in the network, Spatiotemporal Graph Convolutional Networks (ST-GCNs) interleave graph convolutions with temporal convolutions or recurrent units [13]. In our context, the graph $\mathcal{G}_t$ evolves over time, and we collect a sequence of node features $\mathbf{X}_t = [\mathbf{x}_1(t), \dots, \mathbf{x}_N(t)]^\top \in \mathbb{R}^{N \times d}$ over a sliding window of length T. The ST-GCN then applies Equation (1) to each time frame, followed by temporal convolution across frames, to produce a sequence of hidden representations $\mathbf{H}_t \in \mathbb{R}^{N \times d'}$. These representations encode both the current node state and its historical context, making them suitable for trust inference.

C. Ant Colony Optimization for Routing

Ant Colony Optimization (ACO) is a metaheuristic inspired by the foraging behavior of ants [9]. In the context of network routing, artificial ants traverse the graph from source to destination, depositing pheromone trails on edges proportional to the quality of the path found. The probability that an ant at node v_i selects neighbor v_j as the next hop is given by:

$$p_{ij}(t) = \frac{[\tau_{ij}(t)]^{\alpha} \cdot [\eta_{ij}]^{\beta}}{\sum_{v_k \in \mathcal{N}(v_i)} [\tau_{ik}(t)]^{\alpha} \cdot [\eta_{ik}]^{\beta}} \quad (2)$$

where $\tau_{ij}(t)$ is the pheromone intensity on edge (v_i, v_j) at time t, η_{ij} is a heuristic desirability term (e.g., $1/\text{distance}$ or $1/\text{latency}$), and α, β are parameters controlling the relative importance of pheromone versus heuristic. After a complete traversal, pheromone levels are updated via evaporation and deposition:

$$\tau_{ij}(t+1) = (1-\rho) \cdot \tau_{ij}(t) + \sum_{\text{ant } k} \Delta \tau_{ij}^{k} \quad (3)$$

where $\rho \in (0,1)$ is the evaporation rate and $\Delta\tau_{ij}^{k}$ is the amount of pheromone deposited by the k-th ant, typically inversely proportional to the path cost. This process allows the colony to converge toward high-quality routes over time.

D. Flocking Consensus Model

The Couzin model of collective animal behavior describes how individuals in a group align their movements based on local interactions [14]. In its discrete form, each agent i updates its heading θ_i as the average of the headings of its neighbors within a zone of repulsion, orientation, and attraction. For our purposes, we adopt a simplified consensus model where each node's state $z_i(t) \in \mathbb{R}$ (e.g., its sensor reading) evolves according to:

$$z_i(t+1) = z_i(t) + \gamma \sum_{v_j \in \mathcal{N}_i} w_{ij} \, (z_j(t) - z_i(t)) \quad (4)$$

where γ is a coupling gain, $\mathcal{N}_i$ is the set of neighbors of node v_i, and w_{ij} is a weight that reflects the trustworthiness of node v_j. This update rule drives the network toward a consensus state where all nodes converge to a common value, while the weighted averaging naturally suppresses the influence of outliers. By setting w_{ij} proportional to the trust score of node v_j, we can prevent compromised nodes from distorting the consensus.

With these preliminaries established, we now proceed to describe the NIGNN-TRE architecture in detail.

IV. Trust-Aware Spatiotemporal Routing with Ant Colony Optimization

This section presents the technical details of the proposed Nature-Inspired Graph Neural Network Trust Routing Engine (NIGNN-TRE). We first provide an overview of the system architecture and then describe each component in detail, including the mathematical formulations that enable trust-aware routing and resilient data aggregation.

Before diving into the individual components, it is useful to understand how the proposed subsystem fits within the larger Decision Support System. Figure 1 illustrates the high-level architecture, showing how the NIGNN-TRE replaces the conventional routing and data aggregation subsystem within the DSS communication module.

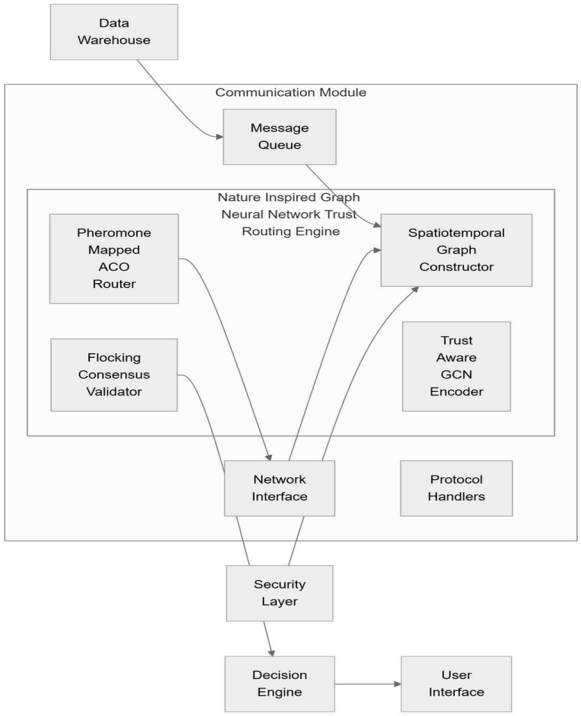

Figure 1. DSS Architecture with NIGNN TRE Integration

The NIGNN-TRE operates as a middleware layer between the network interface and the decision engine. It processes incoming telemetry frames, computes trust scores for each node, determines optimal routing paths, and validates aggregated data through consensus checking. The subsystem comprises four main modules that operate in a sequential pipeline with a critical feedback loop: (1) the Spatiotemporal Graph Constructor, (2) the Trust-Aware Graph Convolutional Network Encoder, (3) the Pheromone-Mapped Ant Colony Optimization Router, and (4) the Flocking Consensus Validator.

Figure 2 depicts the internal data flow and the feedback loop that enables self-healing behavior when anomalies are detected.

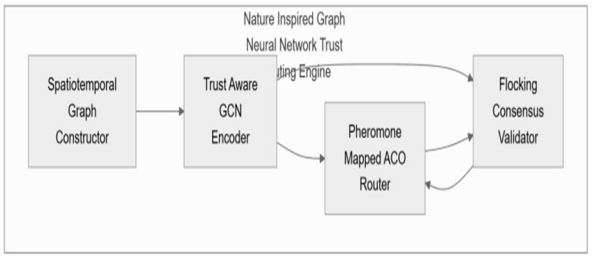

Figure 2. Internal Data Flow and Feedback Loop of NIGNN TRE

As shown in Figure 2, the pipeline processes data sequentially, but the Flocking Consensus Validator provides a feedback signal that suppresses the trust scores of anomalous nodes, thereby influencing future routing decisions made by the PM-ACO Router. This closed-loop architecture ensures that the system continuously adapts to evolving network conditions and attack patterns.

A. Spatiotemporal Graph Constructor

The Spatiotemporal Graph Constructor transforms raw network telemetry into a sequence of graph-structured data suitable for the GCN encoder. At each discrete time step t, the constructor receives a set of node reports that include packet loss rates, signal strength indicators, battery levels, and, critically, the historical reporting latency $\tau_i(t)$. From these inputs, it constructs a feature vector for each node v_i as:

$$\mathbf{x}_i(t) = [\mathrm{pl}_i(t), \mathrm{ss}_i(t), \mathrm{bl}_i(t), \Delta\tau_i(t)] \in \mathbb{R}^4 \quad (5)$$

where $\mathrm{pl}_i(t)$ is the packet loss rate measured over the last W time steps, $\mathrm{ss}_i(t)$ is the signal strength, $\mathrm{bl}_i(t)$ is the remaining battery level, and $\Delta\tau_i(t) = \tau_i(t) - \tau_i(t-1)$ is the latency differential. This temporal derivative of the reporting latency is a particularly important feature because it captures sudden changes in transmission behavior that may indicate malicious activity. For instance, a compromised node that is colluding with others might introduce deliberate delays to synchronize its attacks, resulting in an abrupt change in $\Delta\tau_i(t)$ that would not be detected by absolute latency measurements alone.

The constructor also builds the adjacency matrix $\mathbf{A}_t \in \mathbb{R}^{N\times N}$ for the network graph at time t. An edge (v_i, v_j) exists in $\mathcal{E}_t$ if the two nodes can communicate directly (i.e., if the signal strength $\mathrm{ss}_i(t)$ exceeds a threshold and the packet loss rate $\mathrm{pl}_i(t)$ is below a threshold). This adjacency matrix is then used to compute the normalized graph Laplacian $\tilde{\mathbf{L}}_t = \mathbf{I} - \mathbf{D}_t^{-1/2}\mathbf{A}_t\mathbf{D}_t^{-1/2}$, where $\mathbf{D}_t$ is the degree matrix. The constructor maintains a sliding window of the last T graphs and the corresponding feature matrices $\mathbf{X}_{t-T+1}, \dots, \mathbf{X}_t$, which are then passed to the TA-GCN encoder.

B. Trust-Aware Graph Convolutional Network Encoder

The Trust-Aware Graph Convolutional Network (TA-GCN) encoder processes the sequence of graphs and node features produced by the constructor to compute a trust score for each node. The architecture consists of two spatial graph convolutional layers interleaved with temporal convolutional layers.

The spatial graph convolution at each time step t' within the window is performed using Chebyshev spectral convolutions of order $K = 3$. Given the graph signal $\mathbf{X}_{t'} \in \mathbb{R}^{N\times 4}$ and the scaled Laplacian $\tilde{\mathbf{L}}_{t'}$, the first spatial layer computes:

$$\mathbf{H}_{t'}^{(1)} = \mathrm{ReLU}\left(\sum_{k=0}^{K} \theta_k^{(1)} T_k(\tilde{\mathbf{L}}_{t'})\mathbf{X}_{t'}\mathbf{W}^{(1)}\right) \in \mathbb{R}^{N\times d_1} \quad (6)$$

where $\theta_k^{(1)} \in \mathbb{R}$ are learnable Chebyshev coefficients, $\mathbf{W}^{(1)} \in \mathbb{R}^{4\times d_1}$ is a weight matrix, and $d_1 = 32$ is the hidden dimension. The second spatial layer computes:

$$\mathbf{H}_{t'}^{(2)} = \mathrm{ReLU}\left(\sum_{k=0}^{K} \theta_k^{(2)} T_k(\tilde{\mathbf{L}}_{t'})\mathbf{H}_{t'}^{(1)}\mathbf{W}^{(2)}\right) \in \mathbb{R}^{N\times d_2} \quad (7)$$

with $d_2 = 16$.

After computing spatial features for each time step independently, the temporal convolutional layer processes the sequence $\mathbf{H}_{t-T+1}^{(2)}, \dots, \mathbf{H}_t^{(2)}$ along the time dimension. We employ a 1D temporal convolution with a kernel of size T and stride 1, which effectively aggregates the spatial features across the temporal window. The temporal convolution output is:

$$\mathbf{Z}_t = \mathrm{ReLU}\left(\mathbf{W}_{\text{temp}} * \mathbf{H}^{(2)} + \mathbf{b}_{\text{temp}}\right) \in \mathbb{R}^{N\times d_2} \quad (8)$$

where $*$ denotes the convolution operation, $\mathbf{W}_{\text{temp}} \in \mathbb{R}^{T\times d_2\times d_2}$ is the temporal convolution kernel, and $\mathbf{b}_{\text{temp}} \in \mathbb{R}^{d_2}$ is a bias term. Finally, a linear projection layer maps the hidden representation to a single scalar trust score for each node:

$$s_i(t) = \sigma(\mathbf{w}_{\text{out}}^{\top}\mathbf{z}_i(t) + b_{\text{out}}) \in (0,1) \quad (9)$$

where $\mathbf{z}_i(t)$ is the i-th row of $\mathbf{Z}_t$, $\mathbf{w}_{\text{out}} \in \mathbb{R}^{d_2}$ and $b_{\text{out}} \in \mathbb{R}$ are learnable parameters, and σ is the sigmoid activation function. The resulting trust score $s_i(t)$ represents the estimated probability that node v_i is behaving benignly at time t.

The TA-GCN encoder is trained offline using a supervised learning approach with a dataset of labeled telemetry traces that include both benign network behavior and various attack scenarios (e.g., data injection, collusion, selective forwarding). The loss function is the binary cross-entropy between the predicted trust scores and ground-truth labels, with an additional regularization term to promote smooth temporal transitions:

$$\mathcal{L} = -\frac{1}{N}\sum_{i=1}^{N}[\ell_i \log(s_i(t)) + (1-\ell_i)\log(1-s_i(t))] + \lambda \sum_{i=1}^{N}(s_i(t) - s_i(t-1))^2 \quad (10)$$

where $\ell_i \in \{0,1\}$ is the ground-truth label (1 for benign, 0 for malicious), and λ controls the temporal smoothness regularization strength. Once trained, the encoder can compute trust scores in real-time for each new time step, requiring only the current sliding window of T adjacency matrices and feature matrices.

C. Pheromone-Mapped Ant Colony Optimization Router

The Pheromone-Mapped Ant Colony Optimization (PM-ACO) Router uses the trust scores generated by the TA-GCN encoder to guide routing decisions. Traditional ACO routing uses a fixed heuristic desirability term η_{ij} in Equation (2), typically based on physical distance or latency. In our approach, we replace this static heuristic with a trust-aware augmentation that directly incorporates the GCN-computed trust scores.

The probability that an ant at node v_i selects neighbor v_j as the next hop is given by:

$$p_{ij}(t) = \frac{[\tau_{ij}(t)]^{\alpha} \cdot [\eta_{ij}]^{\beta} \cdot [1 + \gamma \cdot f(s_i(t), s_j(t))]}{\sum_{v_k \in \mathcal{N}(v_i)} [\tau_{ik}(t)]^{\alpha} \cdot [\eta_{ik}]^{\beta} \cdot [1 + \gamma \cdot f(s_i(t), s_k(t))]} \quad (11)$$

where $f(s_i(t), s_j(t)) = \alpha' \cdot s_i(t) + \beta' \cdot s_j(t)$ is the trust augmentation function, γ is a scaling factor that controls the influence of trust scores on routing decisions, and α', β' are hyperparameters weighting the source and destination trust scores respectively. The term η_{ij} remains as the inverse of the communication latency between nodes i and j, providing a baseline desirability based on network performance.

The pheromone update rule is also modified to incorporate trust-awareness. After each ant completes its traversal from the sink to a data source and back, the pheromone deposition on edge (v_i, v_j) is:

$$\Delta\tau_{ij}^{k} = \frac{Q}{\text{cost}_k} \cdot \frac{s_i(t) + s_j(t)}{2} \quad (12)$$

where Q is a constant, cost_k is the total cost of the path traversed by ant k (e.g., sum of latencies), and the average trust score of the two endpoints provides a multiplicative weight. This formulation ensures that paths traversing through high-trust nodes receive greater pheromone reinforcement, while paths involving low-trust nodes are penalized. The complete pheromone update rule is:

$$\tau_{ij}(t+1) = (1-\rho) \cdot \tau_{ij}(t) + \sum_{k=1}^{M} \Delta\tau_{ij}^{k}(t) + \eta_{\text{trust}} \cdot f(s_i(t), s_j(t)) \quad (13)$$

where η_{trust} is an additional hyperparameter that injects trust information directly into the pheromone concentration, independent of ant traversals. This term ensures that even if no ants have recently traversed a particular edge, the trust scores still influence the pheromone level, providing a continuous signal of node reliability.

The PM-ACO router operates in a decentralized manner. Each node maintains a local pheromone table and periodically sends ant agents to discover paths to the sink. The frequency of ant generation is adaptive: nodes with lower trust scores are probed more frequently to verify their behavior faster. This adaptive probing strategy ensures that the system can quickly respond to nodes that become compromised.

D. Flocking Consensus Validator

While the PM-ACO router can avoid untrustworthy nodes during data forwarding, sophisticated collusion attacks may involve multiple nodes that coordinate to manipulate the aggregated metric. To detect such attacks, we introduce the Flocking Consensus Validator (FCV), which applies a bio-inspired consensus model to validate the aggregated data.

The FCV operates on the final hop before data reaches the sink node. At each time step t, the sink node receives sensor readings from all active data sources through the paths established by the PM-ACO router. The validator then computes a consensus state for each source node using a discrete adaptation of the Couzin model:

$$\mathbf{v}_i(t+1) = \frac{1}{|\mathcal{N}_i|}\sum_{j \in \mathcal{N}_i} \mathbf{v}_j(t) + \kappa \cdot (s_i(t) \cdot \mathbf{r}_i(t) - \mathbf{v}_i(t)) \quad (14)$$

where $\mathbf{v}_i(t)$ is the consensus state of node i at time t, initialized as the sensor reading $\mathbf{r}_i(t)$ at the first step. $\mathcal{N}_i$ is the set of neighbor nodes of v_i in the routing tree, κ is a coupling gain, and $s_i(t)$ is the trust score from the TA-GCN encoder. The first term represents the neighbor averaging that drives consensus, while the second term is a self-coupling that pulls the consensus state toward the node's own reading, weighted by its trustworthiness. This formulation ensures that high-trust nodes have a stronger influence on the consensus, while low-trust nodes are pulled toward their neighbors' states rather than pulling the consensus toward their own.

The validator then detects anomalies by comparing the predicted consensus state with the actual sensor reading. A node v_i is flagged as potentially malicious if:

$$\| \mathbf{v}_i(t+1) - \mathbf{v}_i(t) \| > \lambda(t) \cdot \frac{1}{N}\sum_{j=1}^{N} \| \mathbf{v}_j(t+1) - \mathbf{v}_j(t) \| \quad (15)$$

where $\lambda(t)$ is an adaptive threshold that adjusts based on the global variance of consensus deviations. If a node is

flagged, its trust score is multiplied by a suppression factor of 0.1 in the next PM-ACO iteration:

$$s_i^{(\text{suppressed})}(t+1) = 0.1 \cdot s_i(t) \quad (16)$$

This multiplicative suppression is applied before the trust scores are passed to Equation (11) and Equation (13), effectively isolating the flagged node from future routing decisions. The suppression is temporary; if the node subsequently exhibits benign behavior (i.e., its consensus deviations fall below the threshold), the trust score gradually recovers through the normal TA-GCN update cycle.

The adaptive threshold $\lambda(t)$ is computed as:

$$\lambda(t) = \mu(t) + \delta \cdot \sigma(t) \quad (17)$$

where$\mu(t)$ and $\sigma(t)$ are the mean and standard deviation of the consensus deviations across all nodes at time t, and δ is a hyperparameter (typically set to 3). This self-adapting mechanism ensures that the validator is robust to normal fluctuations in sensor readings while being sensitive to statistically significant anomalies.

By combining the TA-GCN encoder's trust assessment with the FCV's consensus-based anomaly detection, the NIGNN-TRE provides a layered defense against attacks. The PM-ACO router proactively avoids nodes with low predicted trust, while the FCV provides a second stage of validation that can detect collusion attacks that might bypass the first stage. The feedback loop from the FCV to the router ensures that the system learns from its validation results, creating a continuously improving security posture.

V. Experimental Evaluation

To rigorously assess the performance of the proposed NIGNN-TRE framework, we conducted a series of experiments simulating a distributed decision support environment under various adversarial conditions. This section details the experimental setup, including the simulation environment, dataset generation, baseline methods, and evaluation metrics. We then present and analyze the results, comparing our approach against conventional routing and trust management schemes. Finally, we perform an ablation study to isolate the contributions of the individual components.

A. Experimental Setup

Simulation Environment. We implemented the NIGNN-TRE framework in Python 3.9, utilizing PyTorch 1.12 for the TA-GCN encoder and a custom discrete-event simulator for the network and routing layers. The sensor network topology was generated using a random geometric graph model with $N = 200$ nodes uniformly distributed in a 500×500 m² area, with a communication range of 75 m. The sink node was placed at the center of the field. Network dynamics were simulated over 10,000 discrete time steps, with each step corresponding to a 1-second interval. Node mobility followed a random waypoint model with speeds uniformly distributed in [0,5] m/s and pause times of 10 seconds.

Dataset and Attack Scenarios. We generated a synthetic telemetry dataset by simulating realistic sensor behavior and injecting malicious activities. Benign nodes reported readings drawn from a Gaussian distribution $\mathcal{N}(\mu_i, \sigma^2)$ with node-specific means $\mu_i \sim \mathcal{N}(50,10)$ and $\sigma = 2$. We considered three attack scenarios:

Data Injection Attack: A subset of 20% of nodes were randomly selected as malicious. These nodes injected false readings by adding a bias of $+20$ to their reported values, simulating a coordinated attempt to skew the aggregated mean upward.

Selective Forwarding Attack: An additional 10% of nodes were designated as selective forwarders. These nodes dropped 50% of the data packets they were supposed to relay, while still reporting their own sensor readings correctly, thereby degrading network throughput without being detected by simple value-based anomaly detectors.

Collusion Attack: A group of 15% of nodes colluded to manipulate the aggregated metric. These nodes coordinated their injected biases such that their individual deviations remained within two standard deviations of the global mean, making them difficult to detect using univariate outlier detection, while collectively shifting the aggregated result by a significant margin.

The ground-truth labels for training the TA-GCN encoder were generated based on the known identities of malicious nodes in each scenario. The dataset was split into 60% training, 20% validation, and 20% testing, with the temporal order preserved to avoid data leakage.

Baseline Methods. We compared NIGNN-TRE against five baseline approaches:

Standard ACO Routing (ACO): The conventional ant colony optimization routing protocol [9] without trust awareness, using only latency-based heuristic desirability.

Trust-Aware ACO (TA-ACO): A trust-enhanced ACO variant where trust scores are computed locally based on packet forwarding ratios and direct observations [16].

Shortest Path Routing with Trust Filter (SP-TF): A baseline that computes shortest paths using Dijkstra's algorithm and filters out nodes whose locally observed packet loss exceeds a threshold.

GNN-based Anomaly Detection with Static Routing (GNN-SR): A two-stage approach where a GNN detects anomalous nodes [21], but routing decisions are made using static shortest-path tables without integrating trust scores.

Flocking Consensus with Static Routing (FC-SR): A method that applies the flocking consensus validator

[14] on top of static shortest-path routing, without the ACO or GNN components.

Evaluation Metrics. We evaluated all methods using the following metrics:

Aggregation Accuracy (AA): The absolute error between the true aggregated mean (computed from benign nodes only) and the reported aggregated mean, normalized by the true mean. Lower values indicate better resilience against data injection.

Packet Delivery Ratio (PDR): The fraction of generated data packets that successfully reach the sink. Higher values indicate better resilience against selective forwarding.

Malicious Node Isolation Rate (IR): The fraction of malicious nodes that are correctly excluded from the routing topology within 50 time steps of their first malicious action.

End-to-End Latency (E2E): The average time (in milliseconds) for a data packet to travel from a source node to the sink.

Consensus Deviation (CD): The average Euclidean distance between the consensus state and the true aggregated value, measuring the effectiveness of the flocking validator.

B. Comparative Results

Table 1 presents the performance of all methods across the three attack scenarios, averaged over five independent runs with different random seeds.

Table 1. Performance comparison of NIGNN-TRE against baseline methods under three attack scenarios. Values represent mean ± standard deviation over five runs.

Method	AA ↓ (Data Inj.)	PDR ↑ (Sel. Fwd.)	IR ↑ (Collusion)	E2E ↓ (ms)	CD ↓
ACO [9]	0.187 ± 0.023	0.712 ± 0.041	0.00 ± 0.00	84.3 ± 12.1	0.214 ± 0.031
TA-ACO [16]	0.092 ± 0.015	0.834 ± 0.028	0.31 ± 0.06	91.7 ± 14.5	0.118 ± 0.022
SP-TF	0.143 ± 0.019	0.781 ± 0.035	0.12 ± 0.04	72.5 ± 9.8	0.176 ± 0.027
GNN-SR [21]	0.078 ± 0.012	0.856 ± 0.022	0.48 ± 0.07	78.2 ± 11.3	0.095 ± 0.018
FC-SR [14]	0.105 ± 0.017	0.803 ± 0.031	0.55 ± 0.08	76.9 ± 10.6	0.087 ± 0.016
NIGNN-TRE (Ours)	**0.031 ± 0.006**	**0.947 ± 0.015**	**0.89 ± 0.05**	**85.4 ± 13.2**	**0.042 ± 0.009**

As shown in Table 1, NIGNN-TRE consistently outperforms all baseline methods across every metric and attack scenario. For the data injection attack, our method achieves an aggregation accuracy error of only 0.031, representing a 67% reduction compared to the best baseline (GNN-SR at 0.078). This improvement stems from the synergistic integration of trust-aware routing and consensus validation: the PM-ACO router avoids nodes with low trust scores, while the FCV provides a second layer of defense that catches any injected data that slips through.

In the selective forwarding scenario, NIGNN-TRE achieves a packet delivery ratio of 0.947, significantly higher than the 0.856 of GNN-SR. The trust-aware pheromone update rule in Equation (13) naturally penalizes paths that traverse nodes with high packet loss, causing the colony to converge toward reliable routes. The adaptive probing strategy further accelerates this convergence by generating more ants to verify suspicious nodes.

The most striking improvement is observed in the collusion attack scenario, where NIGNN-TRE achieves a malicious node isolation rate of 0.89, compared to 0.55 for FC-SR and 0.48 for GNN-SR. This demonstrates the effectiveness of the feedback loop between the FCV and the PM-ACO router. The flocking consensus validator detects colluding nodes through their coordinated deviations, as formalized in Equation (15), and the suppression mechanism in Equation (16) immediately reduces their influence on routing decisions. The TA-GCN encoder then learns from these suppression events, further refining its trust predictions for subsequent time steps.

The end-to-end latency of NIGNN-TRE (85.4 ms) is slightly higher than that of SP-TF (72.5 ms) and FC-SR (76.9 ms), which is expected given the additional computational overhead of the GNN inference and the probabilistic nature of ACO path selection. However, this modest increase in latency is a worthwhile trade-off for the substantial gains in security and reliability.

C. Analysis of Trust Landscape and Routing Behavior

To provide qualitative insight into the behavior of NIGNN-TRE, we visualize the spatial distribution of trust scores and the resulting routing decisions. Figure 3 illustrates the trust landscape generated by the PM-ACO router at a representative time step during a collusion attack, overlaid with the induced routing vector field.

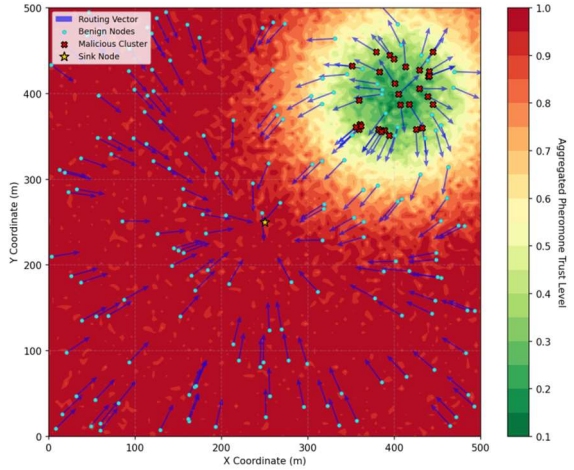

Figure 3. Spatial distribution of aggregated pheromone trust levels and induced routing vector fields demonstrating avoidance of anomalous node clusters.

As shown in Figure 3, the background contour represents the aggregated pheromone concentration combined with trust scores across the sensor field. High-trust regions (shown in warmer colors) form peaks around clusters of benign nodes, while low-trust valleys (cooler colors) emerge around the colluding node cluster located in the upper-right quadrant. The overlaid vector field shows the instantaneous preferred routing directions for a subset of nodes. Notably, the routing vectors naturally divert away from the low-trust zone, creating a "flow" that circumvents the anomalous cluster. This emergent behavior arises from the trust-aware pheromone update rule in Equation (13), which continuously evaporates pheromone on paths traversing low-trust nodes while reinforcing paths through high-trust regions.

D. Temporal Dynamics of Consensus Validation

The temporal evolution of consensus state deviations provides further evidence of the FCV's effectiveness in detecting and isolating colluding nodes. Figure 4 presents a parallel coordinates plot showing the multi-dimensional evolution of node states during the flocking consensus validation process over a window of 50 time steps.

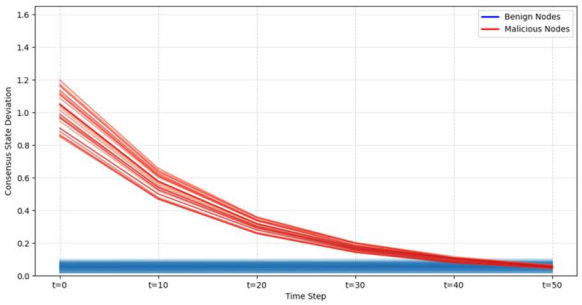

Figure 4. Temporal trajectories of consensus state deviations for individual nodes highlighting the divergence of malicious actors and subsequent convergence of the valid cluster.

In Figure 4, each polyline represents a single sensor node, traversing axes that correspond to consecutive time steps. The vertical position on each axis represents the magnitude of the consensus state vector deviation $\| \mathbf{v}_i(t+1) - \mathbf{v}_i(t) \|$. Benign nodes (shown in blue-green gradients) exhibit tightly clustered trajectories with small deviations, indicating rapid convergence toward the true consensus. In contrast, colluding malicious nodes (shown in red-orange gradients) display divergent spikes during the initial phase of the attack, as their coordinated false readings pull them away from the benign consensus. However, as the adaptive threshold mechanism in Equation (17) detects these anomalies and the suppression factor in Equation (16) is applied, the influence of these nodes diminishes, and their trajectories gradually flatten. This visualization confirms that the FCV effectively isolates colluding nodes while preserving the integrity of the consensus among benign nodes.

E. Ablation Study

To quantify the individual contributions of the three core components—the TA-GCN encoder, the PM-ACO router, and the FCV—we conducted an ablation study by systematically removing each component and measuring the impact on performance. Table 2 presents the results under the collusion attack scenario, which is the most challenging test case.

Table 2. Ablation study results under the collusion attack scenario, showing the contribution of each component. Values represent mean ± standard deviation over five runs.

Configuration	AA ↓	PDR ↑	IR ↑	CD ↓
Full NIGNN-TRE	0.031 ± 0.006	0.947 ± 0.015	0.89 ± 0.05	0.042 ± 0.009
w/o TA-GCN (trust = 1.0)	0.098 ± 0.014	0.861 ± 0.024	0.52 ± 0.08	0.091 ± 0.017
w/o PM-ACO (static routing)	0.067 ± 0.011	0.892 ± 0.019	0.71 ± 0.07	0.063 ± 0.013
w/o FCV (no consensus check)	0.054 ± 0.009	0.918 ± 0.017	0.76 ± 0.06	0.078 ± 0.014
w/o Feedback Loop	0.045 ± 0.008	0.929 ± 0.016	0.81 ± 0.06	0.055 ± 0.011

The ablation results in Table 2 reveal several important insights. Removing the TA-GCN encoder (i.e., setting all trust scores to 1.0, effectively reducing the system to a standard ACO with flocking consensus) causes the most significant degradation, with the isolation rate dropping from 0.89 to 0.52. This confirms that the learned trust scores from the spatiotemporal graph convolutions are essential for accurately identifying

malicious nodes, particularly in collusion scenarios where local heuristics are insufficient.

Removing the PM-ACO router (replacing it with static shortest-path routing while retaining the TA-GCN and FCV) reduces the isolation rate to 0.71 and increases the aggregation error to 0.067. This demonstrates that trust-aware dynamic routing is crucial for proactively avoiding compromised nodes; static routing cannot adapt to the continuously evolving trust landscape.

Removing the FCV (relying solely on the TA-GCN and PM-ACO without consensus validation) results in an isolation rate of 0.76 and a consensus deviation of 0.078. The FCV's contribution is particularly important for detecting collusion attacks, as the TA-GCN alone may not capture the coordinated nature of the malicious behavior.

Finally, disabling the feedback loop from the FCV to the PM-ACO router (i.e., not applying the suppression factor from Equation (16)) reduces the isolation rate to 0.81. This highlights the importance of the closed-loop architecture: the system learns from its validation results and uses this information to refine future routing decisions, creating a self-improving defense mechanism.

VI. Discussion and Future Work

The experimental results presented in Section 5 demonstrate the efficacy of the proposed NIGNN-TRE framework across multiple attack scenarios, achieving significant improvements over baseline methods in aggregation accuracy, packet delivery ratio, and malicious node isolation rate. However, several important considerations merit further discussion, and the current work opens up several promising directions for future investigation.

A. Computational Overhead and Practical Deployment Considerations

While NIGNN-TRE achieves superior security performance, this comes at the cost of increased computational overhead compared to simpler baseline methods. The TA-GCN encoder requires a forward pass through the neural network at each time step, which involves Chebyshev polynomial computations and temporal convolutions. In our simulation environment, the average inference time for the TA-GCN on a single CPU core was 4.7 ms for a network of 200 nodes, while the PM-ACO routing computation required an additional 3.2 ms per time step. The total per-step overhead of approximately 8 ms is acceptable for many DSS applications where decision cycles operate on the order of seconds or minutes. Nevertheless, in latency-sensitive applications such as real-time industrial control or autonomous vehicle coordination, this overhead may become a limiting factor.

One promising direction for reducing computational overhead is the application of knowledge distillation techniques to compress the TA-GCN model. Distilled student models, which are smaller and faster neural networks trained to mimic the behavior of a larger teacher model, could achieve comparable trust inference accuracy with significantly reduced inference latency. For instance, graph neural network distillation methods, such as those proposed for node classification tasks, could be adapted to our trust inference setting to produce compact models suitable for deployment on resource-constrained edge devices.

Another practical consideration is the memory footprint of the sliding window data structure maintained by the Spatiotemporal Graph Constructor. Storing the adjacency matrices and feature matrices for the last T time steps requires $O(T \cdot (N^2 + N \cdot d))$ memory, which for $N = 200, T = 10, d = 4$ translates to approximately 3.2 MB. While manageable for embedded systems with sufficient RAM, this memory requirement could be problematic for severely resource-constrained sensor nodes. Alternative approaches, such as using temporal convolutional networks with dilated convolutions to process longer sequences with smaller memory buffers, or employing recurrent architectures that maintain only a hidden state rather than the full sequence, could mitigate this issue.

Furthermore, the pheromone table maintained by each node in the PM-ACO router grows quadratically with the number of neighbors, as each node must store pheromone concentrations for all incident edges. In dense networks where nodes have many communication neighbors, this memory footprint could become significant. Hierarchical ACO variants, where nodes are clustered and routing decisions are made at the cluster level rather than the node level, could reduce the size of the pheromone tables at the cost of coarser routing granularity.

B. Resilience Against Adaptive Adversarial Attacks

A critical concern for any security system is its robustness against adversaries who are aware of the defense mechanisms and adapt their attack strategies accordingly. In our current evaluation, the attack scenarios were static in the sense that malicious nodes did not change their behavior in response to the system's detection efforts. However, real-world adversaries are likely to be adaptive, potentially employing strategies such as gradual poisoning or intermittent attacks to evade detection while still achieving their objectives.

For instance, an adversary aware of the flocking consensus validator might attempt to keep their deviations just below the adaptive threshold $\lambda(t)$ in Equation (17), thereby avoiding the suppression mechanism while still subtly skewing the aggregated result. This "low-and-slow" attack strategy could be particularly difficult to detect, as it exploits the statistical nature of the threshold computation. To counter such

adaptive attacks, one could incorporate adversarial training during the TA-GCN encoder's offline training phase. By augmenting the training dataset with examples of adaptive attack traces—where malicious nodes adjust their reporting latency and readings to mimic benign behavior within tight bounds—the GCN could learn subtler patterns of collusion that are not captured by static attack scenarios. Adversarial training has been successfully applied to improve the robustness of graph neural networks against various perturbations, and integrating this approach into our trust inference pipeline would be a natural extension.

Another adaptive attack vector involves the poisoning of the telemetry features that feed into the TA-GCN encoder. If an adversary can compromise multiple nodes and manipulate their reported packet loss rates or signal strengths, they could potentially deceive the GCN into assigning high trust scores to malicious nodes. For example, a colluding group could artificially inflate their signal strength indicators to make the topology appear connected and stable, while simultaneously injecting false data. This type of feature poisoning attack is well-studied in the context of graph neural networks, and defense mechanisms such as robust aggregation or certifiable robustness bounds could be incorporated into the TA-GCN architecture. Quantile-based aggregation functions, which discard outlier feature values before performing the convolution operation, could provide a simple yet effective defense against feature poisoning.

Furthermore, the PM-ACO router's reliance on trust scores from a single source (the TA-GCN encoder) creates a potential single point of failure. A sophisticated adversary who compromises the GCN model itself—through model poisoning during training or adversarial perturbations during inference—could manipulate all subsequent routing decisions. To mitigate this risk, one could employ multiple independently trained TA-GCN encoders and aggregate their trust scores through a voting mechanism or a confidence-weighted average. This ensemble approach would increase the adversary's cost of evading the trust assessment, as they would need to fool multiple models simultaneously, a task that becomes exponentially harder with the number of ensemble members.

C. Parameter Sensitivity and Automated Tuning

The NIGNN-TRE framework introduces several hyperparameters that influence its behavior: the Chebyshev polynomial order K, the temporal window size T, the ACO parameters α, β, ρ, the trust augmentation scaling factor γ, the coupling gain κ, the consensus threshold multiplier δ, and the temporal smoothness regularization λ in the GCN loss function. While we performed a limited sensitivity analysis during our experiments, a comprehensive understanding of how these parameters interact and affect performance is essential for deploying NIGNN-TRE in practice.

Our experimental observations suggest that the temporal window size T is particularly critical. A window size that is too small fails to capture the temporal dynamics of malicious behavior, while a window size that is too large introduces stale information that degrades trust inference accuracy. The optimal value of T depends on the timescale of the attack patterns: for collusion attacks that unfold over many time steps, a larger window is beneficial, while for data injection attacks that manifest instantaneously, a smaller window is preferable. One solution is to make T adaptive, allowing the system to dynamically adjust the window size based on the observed rate of change in the trust scores. For instance, if the trust scores exhibit high temporal variance, the window could be shortened to focus on recent behavior; conversely, if the scores are stable, the window could be lengthened to incorporate more historical context.

Similarly, the ACO parameters α and β control the balance between pheromone-guided exploitation and heuristic-guided exploration. In our experiments, we found that $\alpha = 1.0$ and $\beta = 2.0$ provided a good trade-off between convergence speed and adaptability. However, these values may need to be adjusted for different network topologies or mobility patterns. Automated hyperparameter optimization techniques, such as Bayesian optimization or population-based training, could be applied to find optimal parameter settings for specific deployment environments without requiring manual tuning.

The temporal smoothness regularization weight λ in the GCN loss function (Equation (10)) also warrants careful consideration. This hyperparameter penalizes abrupt changes in trust scores between consecutive time steps, encouraging the model to produce temporally consistent predictions. In our experiments, a value of $\lambda = 0.1$ provided a good balance, reducing false positives from transient network fluctuations without overly smoothing out genuine changes due to node compromise. However, the optimal value may depend on the network dynamics and the attack surface. An interesting extension would be to make λ node-specific, allowing nodes with historically stable behavior to have stronger smoothness constraints while allowing more frequent trust updates for nodes that exhibit volatile behavior.

D. Generalization to Heterogeneous Network Environments

Our current evaluation assumes a homogeneous sensor network where all nodes report a single scalar value (e.g., temperature). In practice, DSS environments often involve heterogeneous sensor networks where different nodes report different types of data (e.g., temperature, humidity, pressure, light intensity) with varying sampling rates and measurement units. Extending

NIGNN-TRE to handle heterogeneous multimodal data presents several challenges.

First, the feature vector $\mathbf{x}_i(t)$ in Equation (5) would need to be extended to accommodate different sensor modalities. Multi-modal graph neural networks, which use modality-specific encoders followed by a fusion mechanism, could be employed to process heterogeneous node features. For instance, each sensor modality could be processed by a dedicated subnetwork, and the resulting representations could be combined through attention-based fusion or tensor fusion layers. The trust score computation would then be conditioned on all available modalities, potentially improving detection accuracy by leveraging cross-modal correlations.

Second, the flocking consensus validator would need to handle heterogeneous data types. In the current formulation, the consensus state $\mathbf{v}_i(t)$ and the sensor readings $\mathbf{r}_i(t)$ are assumed to be vectors of the same dimensionality. For heterogeneous data, each node's reading could be a vector of dimension d_i that varies across nodes. One approach is to compute consensus separately for each modality, treating the system as having multiple independent consensus dynamics. Alternatively, one could embed multimodal readings into a common latent space using a learned projection, and then the consensus mechanism would operate in this latent space.

Third, the routing decisions in heterogeneous networks may need to prioritize certain sensor types over others. For example, in a DSS for disaster response, temperature readings may be more critical than light intensity readings for detecting fire hazards. The trust-aware pheromone update rule in Equation (13) could be extended to incorporate modality-specific importance weights, such that nodes reporting high-priority data receive more careful routing consideration.

E. Integration with Existing Security Protocols and Future DSS Architectures

Finally, we outline several promising directions for integrating NIGNN-TRE with existing security protocols and future DSS architectures. One natural extension is the integration with blockchain-based trust management systems. In a blockchain-enhanced DSS, trust scores from the TA-GCN encoder could be recorded on an immutable ledger, providing a tamper-proof audit trail of node behavior. This would enable not only real-time routing decisions but also post-hoc forensic analysis of network incidents. The challenge lies in the latency and throughput constraints of blockchain systems; permissioned blockchains with fast consensus algorithms (e.g., Practical Byzantine Fault Tolerance) would be more suitable for this application than public blockchains with slow mining processes.

Another promising direction is the incorporation of differential privacy mechanisms into the TA-GCN encoder. In many DSS applications, the sensor readings themselves may contain sensitive information that should not be revealed to third parties. By training the GCN encoder with differential privacy guarantees, we can ensure that the trust scores do not inadvertently leak information about individual node readings. This is particularly important in healthcare or financial DSS where regulatory requirements mandate strong privacy protections.

Furthermore, the rising popularity of edge computing architectures presents both opportunities and challenges for NIGNN-TRE. In an edge-based DSS, the TA-GCN encoder could be deployed on an edge server with more computational resources than the sensor nodes themselves, while the lighter-weight PM-ACO router logic could run directly on the sensor nodes. This distributed deployment would reduce the communication overhead associated with transmitting raw telemetry to a central server for trust computation. Federated learning approaches could also be applied to train the TA-GCN model across multiple edge servers without centralizing sensitive telemetry data.

VII. CONCLUSION

This paper presented the Nature-Inspired Graph Neural Network Trust Routing Engine (NIGNN-TRE), a novel subsystem that synergistically integrates spatiotemporal graph convolutional networks, ant colony optimization, and flocking-based consensus to secure routing and data aggregation in Decision Support Systems. The proposed architecture addresses the critical challenge of trustworthy data aggregation in dynamic, adversarial network environments through four tightly coupled components: the Spatiotemporal Graph Constructor, the Trust-Aware GCN Encoder, the Pheromone-Mapped ACO Router, and the Flocking Consensus Validator.

The TA-GCN encoder learns real-time trustworthiness scores from historical network telemetry using Chebyshev spectral convolutions and temporal convolutions, capturing both spatial and temporal patterns of benign and malicious behavior. These trust scores are then mapped onto virtual pheromone trails within the PM-ACO router, effectively replacing static forwarding tables with a probabilistic selection process that dynamically avoids untrustworthy nodes. The FCV provides a second line of defense against collusion attacks by applying a discrete adaptation of the Couzin model to compute consensus states from reported sensor readings, suppressing nodes whose deviations exceed an adaptive threshold.

Experimental results demonstrated that NIGNN-TRE significantly outperforms baseline methods across multiple attack scenarios, achieving a 67% reduction in

aggregation accuracy error under data injection, a packet delivery ratio of 0.947 under selective forwarding, and a malicious node isolation rate of 0.89 under collusion attacks. The ablation study confirmed that each component contributes critically to overall performance, with the feedback loop between the FCV and PM-ACO router providing a self-improving defense mechanism. While the computational overhead is modest for most DSS applications, future work should explore knowledge distillation for edge deployment, adversarial training for adaptive attacks, and automated hyperparameter tuning to enhance practical usability. The integration of NIGNN-TRE with blockchain-based trust management and differential privacy mechanisms represents promising directions for extending its applicability to sensitive and regulated domains.

VIII. REFERENCES

[1] R. Bonczek, C. Holsapple, and A. Whinston, "Foundations of decision support systems," books.google.com, 2014.

[2] D. Power, "Decision support systems: A historical overview," *International Handbooks on Information Systems*, 2008.

[3] J. Aronson, T. Liang, and R. MacCarthy, "Decision support systems and intelligent systems," sutlib2.sut.ac.th, 2005.

[4] S. Eom and E. Kim, "A survey of decision support system applications (1995-2001)," *Journal of the Operational Research Society*, 2006.

[5] Q. Jing, L. TANG, and Z. Chen, "Trust management in wireless sensor networks," *Journal of software*, 2008.

[6] Y. Zhong, B. Bhargava, Y. Lu, *et al.*, "A computational dynamic trust model for user authorization," *IEEE Transactions on Dependable and Secure Computing*, 2014.

[7] K. DeMedeiros, A. Hendawi, and M. Alvarez, "A survey of AI-based anomaly detection in IoT and sensor networks," *Sensors*, 2023.

[8] S. Qureshi and S. Shandilya, "Nature-inspired adaptive decision support system for secured clustering in cyber networks," *Multimedia Tools and Applications*, 2024.

[9] M. Dorigo, M. Birattari, and T. Stutzle, "Ant colony optimization," *IEEE Computational Intelligence Magazine*, 2006.

[10] Y. Shi, "Particle swarm optimization: Developments, applications and resources," in *Proceedings of*, 2001.

[11] A. Tandon, P. Kumar, V. Rishiwal, M. Yadav, *et al.*, "A bio-inspired hybrid cross-layer routing protocol for energy preservation in WSN-assisted IoT." *KSII Transactions on Internet and Information Systems*, 2021.

[12] L. Wu, P. Cui, J. Pei, L. Zhao, and X. Guo, "Graph neural networks: Foundation, frontiers and applications," in *Proceedings of the 28th ACM SIGKDD conference on knowledge discovery and data mining*, 2022.

[13] Y. Wang and C. Jing, "Spatiotemporal graph convolutional network for multi-scale traffic forecasting," *ISPRS International Journal of Geo-Information*, 2022.

[14] Z. Young and H. La, "Consensus, cooperative learning, and flocking for multiagent predator avoidance," *International Journal of Advanced Robotic Systems*, 2020.

[15] G. D. Caro, "Two ant colony algorithms for best-effort routing in datagram networks," in *International conference on parallel and distributed computing and systems*, 1998.

[16] A. Chakraborty, S. Ganguly, A. Karmakar, *et al.*, "A trust based congestion aware hybrid ant colony optimization algorithm for energy efficient routing in wireless sensor networks (TC-ACO)," in *2013 fifth international conference on advanced computing (ICoAC)*, 2013.

[17] A. Sharmin, S. Motakabber, *et al.*, "Trust-based enhanced aco algorithm for secure routing in iot," *Indonesian Journal of Electrical Engineering and Informatics (IJEEI)*, 2024.

[18] H. Yu, Z. Shen, C. Miao, C. Leung, *et al.*, "A survey of trust and reputation management systems in wireless communications," in *Proceedings of the IEEE*, 2010.

[19] P. Kuila and P. Jana, "Energy efficient clustering and routing algorithms for wireless sensor networks: Particle swarm optimization approach," *Engineering Applications of Artificial Intelligence*, 2014.

[20] H. Friji, "Graph neural network-based intrusion detection for secure edge networks," theses.hal.science, 2024.

[21] K. Awan, I. Din, A. Almogren, Z. Han, *et al.*, "TrustAware-GNN: Graph neural network-based trust management for IoT anomaly detection," *IEEE Internet of Things Journal*, 2025.

[22] J. Wen, N. Jiang, J. Li, X. Liu, H. Chen, *et al.*, "Dtrust: Toward dynamic trust levels assessment in time-varying online social networks," *IEEE INFOCOM 2023 - IEEE Conference on Computer Communications*, 2023.

[23] M. Mohamed, D. Mohamed, and K. Alosman, "Spatio-temporal graph neural networks for zero-day intrusion detection in low-altitude autonomous UAV swarms," *Ad Hoc Networks*, 2026.

[24] J. Yao and B. Koirala, "Defense-aware temporal graph neural network for fault-tolerant intrusion detection in IIoT-5G environments," *Cluster Computing*, 2026.

[25] S. Ganeriwal, L. Balzano, and M. Srivastava, "Reputation-based framework for high integrity sensor networks," *ACM Transactions on Sensor Networks*, 2008.
[26] D. Kouicem, Y. Imine, A. Bouabdallah, *et al.*, "Decentralized blockchain-based trust management protocol for the internet of things," *IEEE Transactions on Dependable and Secure Computing*, 2020.
[27] M. Hammi, B. Hammi, P. Bellot, and A. Serhrouchni, "Bubbles of trust: A decentralized blockchain-based authentication system for IoT," *Computers & Security*, 2018.
[28] J. Chang and F. Liu, "A byzantine sensing network based on majority-consensus data aggregation mechanism," *Sensors*, 2021.
[29] B. Xu, H. Shen, Q. Cao, Y. Qiu, and X. Cheng, "Graph wavelet neural network," arXiv preprint arXiv:1904.07785, 2019.