Graphsafe: Intelligent Intrusion Detection System with Explainable AI

Saumya Yadav[1], Anjali Singh[2], Prerana Upadhyay[3], Professor Prakash Khelage[4]
[1,2,3] *IT, Usha Mittal Institute of Technology SNDT Women's University Mumbai - 400049, India*
[4]*Professor, IT, Usha Mittal Institute of Technology SNDT Women's University Mumbai - 400049, India*

Abstract—**Contemporary cyber threats leverage the relational complexity of networks, frequently bypassing traditional Intrusion Detection Systems (IDS) that process traffic data independently. This thesis introduces a Graph Neural Network-based Intrusion Detection System to counter these shortcomings by representing network entities and their interactions as graphs. Employing Graph Neural Networks (GNNs), GNN-IDS captures the static structure and temporal evolution of network activity, providing a broader perspective than standard techniques. The system combines historical attack graphs with real-time traffic, utilizing Graph Convolutional Networks (GCN), Graph Attention Networks (GAT), and an edge-weighted GCN variant (GCN- EW) to detect intricate attack signatures. It further enhances resilience against adversarial interference and offers interpretable results through uncertainty quantification and attention-driven analysis. Evaluated using synthetic datasets and the CICIDS- 2017 benchmark, GNN-IDS achieves notable accuracy, robustness against tampering, and clarity in tracing intrusion routes. These outcomes demonstrate the power of GNNs to strengthen intrusion detection by exploiting network connectivity, laying groundwork for more effective and transparent cybersecurity solutions. This research elevates IDS performance and invites exploration into graph-centric security approaches.**

Index Terms—**Intrusion Detection System (IDS), Graph Neural Networks (GNNs), Network Security, Cybersecurity, Attack Graphs, Anomaly Detection, Adversarial Resilience, Explainability, Network Traffic Analysis, Deep Learning.**

I. INTRODUCTION

In 2025, networks endure over 2,200 cyberattacks daily [27], a stark reminder that intelligent security is non-negotiable. Intrusion Detection Systems (IDS) form the frontline, vigilantly scanning traffic to detect unauthorized actions or anomalies where traditional IDS fall short. Built on signature matching or anomaly detection, they view flows in isolation, missing the networked patterns that today's threats weave, exposing systems to risk. Intrusion Detection Systems (IDS) are vital for monitoring network activities and identifying unauthorized or anomalous behaviors. Conventional IDS approaches, such as signature-based and anomaly-based detection, typically analyze individual network flows or log files in isolation. This compartmentalized analysis overlooks the complex, interrelated nature of modern network infrastructures, leading to reduced detection accuracy and increased false positive rates. To address these limitations, this paper proposes GNN-IDS, an innovative framework that employs Graph Neural Networks (GNNs) to capture both the structural and dynamic aspects of network environments. GNNs are particularly effective in modeling data with inherent graph-like structures, such as communication networks, where nodes represent devices and edges denote connections. By learning from both static attack graphs and dynamic network measurements, GNN- IDS provides a holistic view of the network, enabling more accurate and explainable intrusion detection.

II. LITERATURE SURVEY

A. *Signature-Based Intrusion Detection Systems (SIDS)*

Key Research:

- Denning (1987) [1] introduced one of the first IDS models, emphasizing rule-based anomaly detection.
- Snort (Roesch, 1999) [2] and Suricata (Vermilion, 2010) became widely adopted open-source IDS solutions using signature-based detection.

Key Findings:

- Signature-based IDS provides high accuracy

for known threats [2].
- Low computational cost compared to anomaly-based detection [1].

Gaps in Knowledge:
- Fails to detect zero-day attacks and new malware variants [2].
- Requires constant rule updates, leading to high maintenance overhead [1].

B. *Anomaly-Based Intrusion Detection Systems (AIDS)*

Key Research:
- Lippmann et al. (2000) [3] conducted the DARPA IDS Evaluation, highlighting the advantages of statistical anomaly detection.
- Tavallaee et al. (2009) [4] introduced the NSL-KDD dataset, improving evaluation for anomaly-based IDS.

Key Findings:
- Effective for detecting unknown attacks by identifying deviations from normal traffic behavior [3].
- Can be applied to real-time monitoring in dynamic environments [4].

Gaps in Knowledge:
- High false positive rates due to difficulties in distinguishing legitimate behavior changes from intrusions [4].
- Poor performance in large-scale networks due to high computational costs [3].

C. *Machine Learning-Based IDS*

Key Research:
- Bhuyan et al. (2014) [5] reviewed supervised and unsupervised ML techniques for intrusion detection.
- Vinayakumar et al. (2019) [6] explored deep learning (DL)-based IDS, demonstrating improved accuracy over traditional ML models.

Key Findings:
- ML models, particularly Random Forest, SVM, and Deep Learning, significantly improve detection accuracy [6].
- Feature engineering and dataset quality play a critical role in performance [5].

Gaps in Knowledge:
- Labeling large datasets for supervised ML is costly and time-consuming [6].
- ML-based IDS is vulnerable to adversarial attacks, where small input modifications can evade detection [5].

D. *Graph-Based Intrusion Detection Systems (GIDS)*

Key Research:
- Liu et al. (2020) [7] introduced a graph-based IDS that models network entities as a graph, improving threat detection.
- Neil et al. (2009) [3] explored attack graph ranking, enhancing the prioritization of security threats.

Key Findings:
- Graph-based models improve context-aware intrusion detection by analyzing relationships between network entities [7].
- Useful for detecting advanced persistent threats (APTs) and attack path analysis [3].

Gaps in Knowledge:
- Constructing real-time network graphs is computationally expensive [7].
- Susceptible to graph adversarial attacks, where attackers manipulate graph structures [3].

E. *Graph Neural Networks (GNN)-Based IDS*

Key Research:
- Jmal et al. (2023) [9] proposed SPGNN-API, a transfer- able GNN model for identifying attack paths.
- Zou et al. (2021) [8] introduced TDGIA, analyzing adversarial vulnerabilities in GNN-based IDS.
- Van Langendonck et al. (2024) [10] developed PPT-GNN, a pre-trained spatiotemporal GNN for near real-time IDS.

Key Findings:
- GNNs effectively capture dynamic network structures and long-range dependencies [9].
- Outperforms traditional ML and deep learning models in handling network topology changes [10].

Gaps in Knowledge:
- Adversarial robustness remains a challenge, as GNNs can be misled by subtle graph modifications [8].
- Scalability issues in large, high-speed networks due to the complexity of graph computations [10].

III. EXISTING SYSTEMS

Existing intrusion detection systems (IDS) employ various techniques to identify cyber threats. These techniques can be categorized as follows:

A. Signature-Based IDS (SIDS)

Examples: Snort, Suricata

Methodology: Compares network traffic with predefined at- tack signatures.

Limitations: Ineffective against zero-day attacks and unknown threats.

B. Anomaly-Based IDS (AIDS)

Examples: Machine learning-based models (Random Forest, SVM, Deep Learning)

Methodology: Flags deviations from normal network behavior.

Limitations: High false positive rates, as non-malicious anomalies can be misclassified.

C. Specification-Based IDS

Examples: Protocol-based intrusion detection models Methodology: Uses pre-defined behavioral rules to identify abnormal activity.

Limitations: Rule maintenance is difficult and requires manual updates.

D. Heuristic-Based IDS

Examples: Hidden Markov Models (HMM), Bayesian networks

Methodology: Uses probability-based models to predict intrusions.

Limitations: Computationally intensive and lacks real-time adaptability.

E. Hybrid IDS

Examples: AI-powered IDS integrating multiple techniques Methodology: Combines signature-based and anomaly-based detection for enhanced accuracy.

Limitations: Resource-intensive and complex to implement in large-scale networks.

Despite these methods, traditional IDS approaches struggle to handle sophisticated multi-stage attacks and evasive mal- ware techniques. To overcome these challenges, graph-based intrusion detection has emerged as a promising alternative.

F. Proposed GNN-IDS Framework

The GNN-IDS framework improves intrusion detection by modeling network traffic as a graph, where:

- Nodes represent devices, processes, or data packets.
- Edges denote communication links, network flows, or dependencies.
- Node features encode attributes such as IP addresses, timestamps, and security logs.
- Edge features capture connection strength, packet types, and protocol usage.

G. Graph Neural Network Architecture

The system is built using PyTorch Geometric (PyG) and follows a structured pipeline:

1) Graph Construction:

- Converts raw network data into structured graphs using adjacency matrices.
- Constructs attack graphs based on past intrusion patterns.

2) Feature Encoding:

- Converts network activities into low-dimensional embed- dings.
- Enriches data with neighbor-aware transformations.

3) Graph Convolutional Layers:

- Utilizes Graph Attention Networks (GAT) and Graph Convolutional Networks (GCN).
- Implements message passing mechanisms for learning attack signatures.

4) Intrusion Classification:

- Categorizes network activities as benign or malicious based on graph embeddings.
- Incorporates uncertainty estimation to resist adversarial attacks.

5) Explainability & Forensics:

- Highlights critical nodes in attack graphs for security analysts.
- Uses attention heatmaps to improve IDS interpretability.

H. Experimental Evaluation

The model is tested on:

- Synthetic datasets with simulated attack patterns.
- CICIDS-2017 dataset, a real-world benchmark for intrusion detection.

Performance metrics include:

- Detection Accuracy (for classification effectiveness).
- False Positive Rate (FPR) (to minimize misclassifications).
- Adversarial Robustness (evaluated through attack simulations).

IV. FRAMEWORK

The Graph Neural Network-based Intrusion Detection Sys- tem (GNN-IDS) is structured to analyze network traffic dynamically and identify anomalies by leveraging graph-based modeling. Unlike conventional IDS approaches, which analyze individual network flows in isolation, GNN-IDS models network entities and their interactions as a graph to capture hidden relationships and improve detection accuracy.

A. Graph-Based Representation of Network Traffic

Network traffic is transformed into a graph structure, where: Nodes represent network entities (e.g., IP addresses, devices, processes). Edges represent interactions (e.g., network flows, data transmissions, attack paths). Features associated with nodes and edges include timestamps, protocol types, packet sizes, and security logs.

B. Graph Construction and Feature Encoding

The system converts raw network logs into structured graph data using adjacency matrices and attack graph generation methods. Feature encoding techniques are applied to transform network traffic attributes into vector embeddings for better learning.

C. Graph Neural Network Processing

The system employs Graph Convolutional Networks (GCN) and Graph Attention Networks (GAT) for: Message Passing: Learning relationships between nodes. Feature Aggregation: Capturing long-range dependencies in network traffic. Different variants such as Edge-Weighted Graph Convolutions (GCN-EW) improve anomaly detection by integrating edge- level attributes.

D. Anomaly Detection and Intrusion Classification

The processed graph is fed into a classification model, which labels network activities as benign or malicious. The model leverages uncertainty estimation to reduce false positives and provide confidence scores for predictions.

E. Explainability and Forensic Analysis

The system uses attention heatmaps to highlight critical nodes and edges in detected attack paths. Security analysts can trace how an attack propagates within the network, improving forensic capabilities.

V. PROPOSED SYSTEM

To overcome the limitations of traditional IDS, the pro- posed GNN-IDS framework leverages Graph Neural Networks (GNNs) to provide adaptive, scalable, and explainable intrusion detection.

A. Graph-Based Intrusion Detection

Unlike signature-based IDS, which relies on predefined attack signatures, GNN-IDS can detect novel attacks by analyzing structural anomalies in network graphs. The system learns attack patterns from both historical attack graphs and real-time network flows.

B. Adversarial Resilience

Attackers often modify network traffic to evade detection. GNN-IDS employs adversarial defense mechanisms, making it robust against evasion techniques like polymorphic malware and encrypted malicious traffic.

C. Real-Time Anomaly Detection with Edge Features

The framework incorporates edge-weighted GNN models that analyze packet interactions, session durations, and net- work topology changes in real time. This improves detection accuracy while reducing computational overhead.

D. Self-Supervised Learning for Zero-Day Attacks

Traditional ML-based IDS requires large labeled datasets, which are costly and time-consuming to create. The proposed system uses self-supervised learning, where the model learns patterns without needing explicit attack labels, making it effective against previously unseen threats.

VI. METHODOLOGY

This study presents a Graph Neural Network-based Intrusion Detection System designed to enhance cybersecurity through structured data processing, attack graph modeling, and machine learning techniques. The methodology consists of several key stages, including data preprocessing, attack graph construction, synthetic data generation, model training, evaluation, and robustness assessment.

A. Data Collection and Preprocessing

The datasets used in this research include a synthetically generated dataset and a real-world dataset based on CIC- IDS2017 network traffic. The CIC-IDS2017 dataset was chosen for its comprehensive representation of modern cyberattack scenarios. The GNN-IDS employs 78 features, consistent with the CIC-IDS2017 dataset, capturing essential network traffic patterns necessary for detecting intrusions. The selection of these features is based on their relevance to intrusion detection, such as network flow characteristics and protocol behaviors. Although the source does not specify explicit strategies for handling missing values, min-max normalization is

applied to ensure uniform data scaling.
The dataset is divided into training, validation, and testing sets, maintaining class distribution through a stratified split. The training set comprises 3000 graphs (21,000 nodes), while the validation and testing sets contain 1000 graphs (7,000 nodes) each. Each graph includes seven vulnerable nodes, with at most one node being compromised at a time, ensuring a structured and systematic approach to identifying intrusion attempts.

B. Feature Engineering

Table I outlines the feature set employed in our GNN-IDS implementation, derived from both static network properties and dynamic traffic patterns. Following data preprocessing, we establish the graph structure by defining the adjacency matrix A, which encapsulates network interactions. This construction is mathematically detailed in Equation (1):

$$A_{vu} = \begin{cases} \phi(t_{vu}), & \text{if a flow exists between nodes } v \text{ and } u \\ 0, & \text{otherwise} \end{cases} \tag{1}$$

In this definition, A_{vu} represents the edge weight between nodes v and u, modulated by the function $\phi(t_{vu}) = \frac{1}{1+e^{-t_{vu}}}$. This sigmoid-based weighting reflects the temporal proximity of interactions, assigning higher values to recent flows. Subsequently, we aggregate raw node features into a unified vector x_v, as expressed in Equation (2):

$$x_v = \psi(F_v) = \sum_{k=1}^{K} w_k f_{vk} \tag{2}$$

Here, F_v comprises $K = 78$ features extracted from the CICIDS-2017 dataset, with each feature f_{vk} weighted by a learned coefficient w_k. This dual methodology—combining a time-sensitive graph structure with weighted feature aggregation—enhances the GNN's capacity to discern patterns from both topological and attribute-based data.

TABLE I: FEATURE ENGINEERING SPECIFICATIONS

Category	Features	Preprocessing
Node Features	Host metrics, Traffic statistics	Normalization, One-hot encoding
Edge Features	Flow duration, Packet rate	Standard scaling
Temporal Features	Time windows, Flow intervals	Log transformation

C. Attack Graph Construction

To model network security states and attack propagation paths, the study utilizes the MulVAL attack graph generator. The attack graph is structured with nodes representing network entities (hosts, assets, and vulnerabilities) and edges encoding logical dependencies that describe how attacks can propagate through the system. The attack graph includes fact nodes, action nodes, and privilege nodes, encoding the relationships between vulnerabilities and exploitability conditions. While the attack graph itself remains static, dynamic behavior is introduced by incorporating real-time measurements, which are associated with privilege nodes, ensuring the system captures evolving network activity. The generated attack graph is then encoded by extracting and tokenizing node relationships, enabling the model to learn attack patterns effectively.

D. Synthetic Data Generation

Synthetic data plays a crucial role in the development of Graph Neural Network-based Intrusion Detection Systems (GNN-IDS) due to challenges associated with real-world cybersecurity datasets. Organizations often restrict access to net-work logs due to security concerns and regulatory constraints, limiting the availability of diverse attack data. Synthetic datasets enable researchers to model cyber threats without exposing sensitive information.
Publicly available datasets frequently lack essential infrastructure details, such as network topologies, asset relation- ships, and privilege escalation paths—critical components for constructing attack graphs in GNN-IDS. Synthetic data bridges this gap by incorporating these missing elements, ensuring a comprehensive evaluation of detection models.
Furthermore, synthetic datasets provide precise control over attack scenarios, evolving threat patterns, and data distributions, allowing for rigorous model testing. To maintain realism, these datasets simulate benign and malicious activities using probability distributions like Gaussian and Poisson models. Attack actions are designed to impact system measurements, replicating real-world behaviors. Additionally, synthetic data helps mitigate dataset imbalances by generating attack samples in controlled proportions, improving detection accuracy.
While real-world data remains valuable for validation, synthetic datasets enhance research by offering diverse attack scenarios, adaptable testing environments, and greater resilience against adversarial techniques.

TABLE II: PARAMETERS FOR DATASET 1: BENIGN AND MALICIOUS (NON-PERSISTENT, PERSISTENT) SCENARIOS, WITH $m1$ (GAUSSIAN: μ, σ), $m2$ (POISSON: λ), AND $m3$ ACROSS 5000 GRAPHS.

Scenario	N	m_1 (N)	m_2 (P)	m_3
Benign	1500	0.3, 0.2	3.0	$\sqrt{\lvert m_1\rvert} + 0.5m_2 + 0.5$
Non-Pers.	500	0.3, 0.2	$U(1, 5)$	$\sqrt{\lvert m_1\rvert} + 0.5m_2 + 0.5$
Pers.	500	$U(0.1, 0.5)$, 0.1	3.0	$\lvert m_1\rvert + 0.5m_2 + 0.5$

E. Performance Metrics

The system's performance is evaluated using the following metrics:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (3)$$

$$Precision = \frac{TP}{TP + FP} \quad (4)$$

$$Recall = \frac{TP}{TP + FN} \quad (5)$$

$$F1\text{-}score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \quad (6)$$

where TP, TN, FP, and FN represent True Positives, True Negatives, False Positives, and False Negatives, respectively.

F. Robustness Evaluation

To test the resilience of the GNN-IDS against adversarial perturbations, we inject Gaussian noise into the features of malicious nodes, as previously noted. This process is defined in Equation (7):

$$' = x_v + \eta \cdot \mathrm{N}(0, \sigma^2) \ (5) \quad (7)$$

Here, x'_v is the perturbed feature vector for node v, obtained by adding noise scaled by $\eta = 0.1$ to the original x_v, with N $(0, \sigma^2)$ sampled at variances $\sigma^2 = 0.01$ and 0.05. This simulates subtle manipulations attackers might use to evade detection. By comparing precision, recall, and F1-score before and after perturbation, we quantify the model's robustness, ensuring it remains effective under adversarial conditions.

VII. IMPLEMENTATION

This section details the implementation of our GNN-IDS system, including the neural network architecture, training process, and optimization techniques.

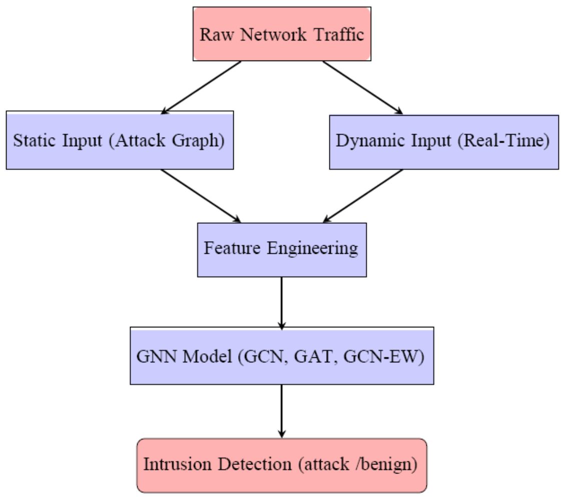
Raw Network Traffic
Static Input (Attack Graph)
Dynamic Input (Real-Time)
Feature Engineering
GNN Model (GCN, GAT, GCN-EW)
Intrusion Detection (attack /benign)

Fig. 1. Proposed architecture for intrusion detection using Graph Neural Networks

A. GNN-IDS Architecture

The architecture of the proposed Graph Neural Network- based Intrusion Detection System (GNN-IDS) is designed to effectively capture complex attack patterns within network traffic. It consists of three primary GNN variants: Graph Convolutional Network (GCN), Graph Attention Network (GAT), and Graph Convolutional Network with Edge Weights (GCN-EW), each tailored to enhance the system's ability to analyze structured attack data. The models take as input an attack graph where nodes represent network events or vulnerabilities and edges define their relationships. The node features are derived from a combination of predefined attack signatures and real-time network measurements.

The GCN architecture consists of two hidden layers, each utilizing graph convolution operations to aggregate neighbor- hood information. The input features, comprising both node statements (K) and real-time measurements (D), pass through these layers with ReLU activation, ensuring non-linearity and efficient feature extraction. The final representation is projected to a single output layer with a Sigmoid activation function, enabling binary classification of attack scenarios.

In contrast, the GAT model incorporates an attention mechanism to dynamically assign importance to neighboring nodes during feature aggregation. It employs multi-head attention with four attention heads, allowing the model to learn multiple perspectives of node interactions. The attention coefficients are computed using learnable parameters, and the outputs from all attention heads are concatenated before passing through the final classification layer. This design enhances the model's ability to focus on critical network connections while filtering out less relevant interactions.

The GCN-EW variant extends the standard GCN by incorporating edge weights that are learned during training. Instead of treating all connections equally, the model assigns different importance values to edges, optimizing the influence of specific relationships within the graph. This is particularly useful in intrusion detection, where certain attack paths may be more indicative of malicious activity. The edge-weighted adjacency matrix refines information propagation, allowing the model to better capture hierarchical attack patterns.

Across all three architectures, the hidden layer dimensions are set to 20, striking a balance between computational efficiency and expressive power. The models are trained using back propagation, with optimization techniques ensuring convergence to an optimal decision boundary. By leveraging GNNs, the system effectively identifies attack patterns that traditional machine learning approaches might overlook, making it a powerful tool for intrusion detection.

The model architecture is defined as:

$$h_i^{(l+1)} = \sigma\Big(\sum_{j \in \mathrm{N}_i} \alpha_{ij} W^{(l)} h_j^{(l)}\Big) \tag{8}$$

where $h_i^{(l)}$ represents the node features at layer l, α_{ij} are attention coefficients, and $W^{(l)}$ are learnable parameters.

B. Encoding Attack Graphs

The attack graph encoding process converts textual information associated with graph nodes into structured feature representations. This transformation enables GNNs to analyze security threats by learning relationships between nodes in the attack graph.

The encoding begins with an initialization step, where a corpus of unique words is constructed from all node descriptions. Each node in the attack graph contains a set of statements representing its properties. The algorithm extracts individual words from these statements and adds them to a growing vocabulary if they are not already included. This ensures that the corpus captures all unique terms across the dataset.

Once the vocabulary is built, the algorithm constructs a binary feature matrix. Each node is assigned a feature vector of length equal to the corpus size, where the presence of a word in the node's statement is marked as 1, and its absence is marked as 0. This one-hot encoding scheme facilitates effective analysis of attack graphs using GNNs.

Algorithm: Attack Graph Encoding

```
Algorithm 1 Encoding Attack Graph for GNN-Based Intrusion Detection
Require: Set of nodes $V$, edges $E$, and statements $S_v$ for
    each $v \in V$
Ensure: Encoded attack graph $G = (V, E, F)$ with feature
    matrix $F$
 1: Initialize index $i \leftarrow 0$
 2: Initialize empty corpus $C$
 3: for each node $v \in V$ do
 4:     for each word $w \in S_v$ do
 5:         if $w \notin C$ then
 6:             $C_i \leftarrow w$
 7:             $i \leftarrow i + 1$
 8:         end if
 9:     end for
10: end for
11: Define corpus size $D \leftarrow |C|$
12: for each node $v \in V$ do
13:     Initialize feature vector $f_v$ of length $D$
14:     for $i \leftarrow 0$ to $D - 1$ do
15:         if $C_i \in S_v$ then
16:             $f_{vi} \leftarrow 1$
17:         else
18:             $f_{vi} \leftarrow 0$
19:         end if
20:     end for
21: end for
```

C. Model Training and Optimization

To train the GNN-IDS effectively, we utilize a weighted cross-entropy loss function to tackle the disparity between benign and malicious samples, as previously discussed. This loss is mathematically expressed in Equation (9):

$$L = -\frac{1}{N}\sum_{v=1}^{N}[\beta y_v \ln(\hat{y}_v) + (1 - y_v)\ln(1 - \hat{y}_v)] \tag{9}$$

In this expression, L represents the average loss computed across N nodes in the graph. The term y_v denotes the true label for node v (where 0 indicates benign and 1 indicates malicious), while $\hat{y}_v$ is the predicted probability of node v being malicious, as output by the model. The weighting factor β, set to 2 based on the observed class distribution, increases the penalty for errors in classifying malicious nodes, compensating for their relative rarity in the dataset. This loss function is paired with the Adam optimizer, configured with a learning rate of 1.0×10^{-3}, and mini-batch training with a batch size of 30. Together, these elements promote stable convergence and emphasize the accurate detection of intrusions.

D. *Inference*

Intrusion detection primarily involves a classification task, which can be either binary (distinguishing between benign and malicious activity) or multi-class (identifying specific attack types). In our approach, the GNN model assigns probability scores to nodes within the network graph, determining whether a particular host is under attack and identifying the malicious action associated with it.

a) Uncertainty: Uncertainty plays a critical role in ma- chine learning-based intrusion detection systems, especially when attackers attempt to evade detection by imitating normal behavior. To address this, we compute probability scores for both benign and malicious classifications at the node level. These scores indicate the model's confidence in its predictions. Additionally, we analyze how varying probability thresholds affect classification accuracy, providing insights into the robustness of our approach against adversarial tactics.

b) Explainability: Understanding the reasoning behind model predictions is essential for effective intrusion detection. Unlike conventional black-box IDS solutions, our GNN-based system leverages network topology to determine the influence of neighboring nodes and features on classification decisions. We employ *GNNEXPLAINER* [18] to evaluate the significance of different node relationships and attributes, enhancing the interpretability of our model.

Furthermore, the proposed IDS incorporates detailed net- work characteristics, such as attack actions and system vulnerabilities, into node representations. This integration enables the system to not only detect anomalies but also trace the underlying attack scenario, improving transparency and making the model's decisions more reliable.

VIII. EXPERIMENTAL RESULTS

A. *Evaluation of Graph-Based Models*

This section presents a comparative analysis of graph-based models, including Neural Network (NN), Graph Convolutional Network (GCN), Graph Convolutional Network with Edge Weights (GCN-EW), and Graph Attention Network (GAT), evaluated on both Dataset1 and Dataset2. The models were assessed using key performance metrics such as accuracy, precision, recall, F1-score, and Area Under the Curve (AUC).

B. *Robustness and Model Generalization*

For Dataset1, GCN demonstrated the highest accuracy of 90.27%, followed by GCN-EW at 88.91%. NN and GAT achieved lower accuracy scores of 73.84% and 68.00%, respectively. However, precision and recall analysis provided further insights into model performance. GCN attained the

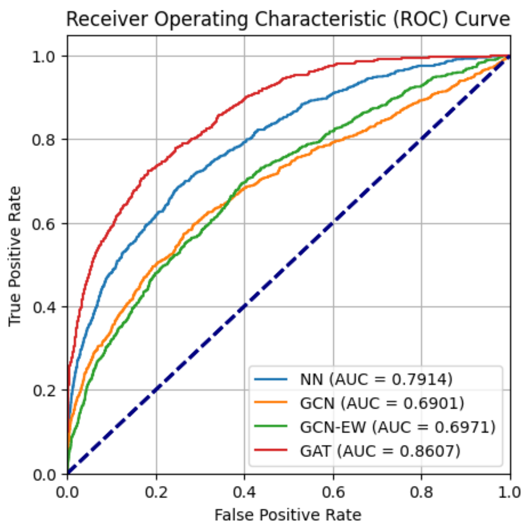

Fig. 2. *

(a) ROC Curve for Dataset 1

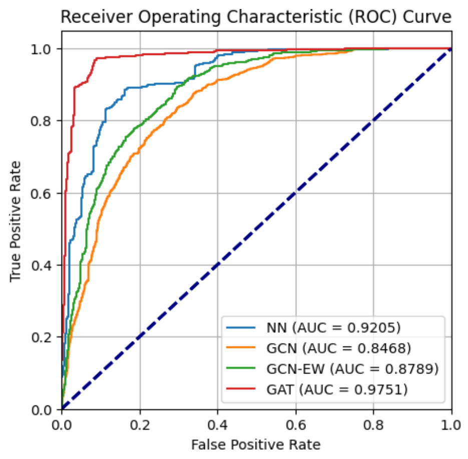

Fig. 3. *

(b) ROC Curve for Dataset 2

Fig. 4. ROC Curves for Dataset 1 and Dataset 2.

highest precision of 83.74%, indicating a low false positive rate (FPR), while GAT achieved the highest recall of 75.62%, demonstrating strong positive instance detection despite a higher FPR. The F1-score analysis revealed that NN (59.05%) and GCN-EW (57.19%) maintained competitive performance, whereas GCN had a lower F1-score of 51.11% due to its reduced recall. AUC analysis further highlighted that GAT achieved the highest value of 0.8607, indicating robust clas- sification capabilities, whereas GCN (0.6901) and GCN-EW (0.6971) exhibited moderate performance.

For Dataset2, a similar trend was observed, with GAT outperforming other models. The NN model achieved an F1-score of 0.6360, recall of 0.8213, and precision of 0.6322, while GCN attained an F1-score of 0.6090, recall of 0.7662, and precision of 0.6105. The AUC score for GCN was 0.8468, suggesting moderate classification effectiveness. The GCN- EW model showed slight improvements over GCN, reaching an F1-score of 0.6321, recall of 0.7910, and an AUC value of 0.8789. However, GAT emerged as the most effective model across both datasets, achieving an F1-score of 0.8057, recall of 0.9342, and precision of 0.7526. It also recorded the highest AUC value of 0.9751 and the lowest FPR of 0.1044, signifying its superior ability to distinguish between positive and negative instances.

To further evaluate model reliability, robustness analysis was conducted by subjecting the models to varied testing conditions. For Dataset1, GCN exhibited the lowest FPR (0.0013), ensuring fewer incorrect classifications, but had the highest false negative rate (FNR) of 0.9614, indicating difficulty in capturing positive cases. Conversely, GAT achieved the lowest FNR (0.1486), demonstrating superior detection of positive instances but with a higher FPR (0.3390). These trade-offs indicate that model selection should be guided by application-specific requirements, where either high precision or high recall may be prioritized.

For Dataset2, robustness evaluation revealed that NN retained an F1-score of 0.6360, with only a minor reduction in its AUC to 0.9175. Similarly, GCN exhibited slight variations, achieving an F1-score of 0.6099, recall of 0.7683, and an AUC of 0.8465. The GCN-EW model remained stable, with a slight decrease in its AUC value to 0.8786 while maintaining an F1-score of 0.6319 and recall of 0.7903. GAT continued to outperform other models, with an F1-score of 0.8057 and only a marginal drop in its AUC to 0.9747. The consistency of these results across multiple datasets highlights GAT's strong generalization ability.

C. *Interpretability and Feature Importance Analysis*

To enhance model interpretability, explainability techniques were applied to both datasets. Using GNNExplainer, key features contributing to model predictions were identified. For Dataset1, the probability scores generated by models ranged from 0.45 to 0.49, indicating moderate classification confidence. Feature importance analysis highlighted critical attributes influencing predictions, while graph structural anal- ysis identified key nodes and edges affecting classification outcomes.

For Dataset2, the attention mechanism in GAT enabled superior feature attribution analysis, allowing the identification of influential nodes and edges in the graph. GCN and GCN- EW, while effective, lacked the interpretability offered by GAT's attention-based mechanism. This visualization further reinforced the effectiveness of GAT by demonstrating its ability to accurately assign importance to critical features.

D. *Summary of Key Findings*

The evaluation across both datasets highlights key trade-offs in model performance:

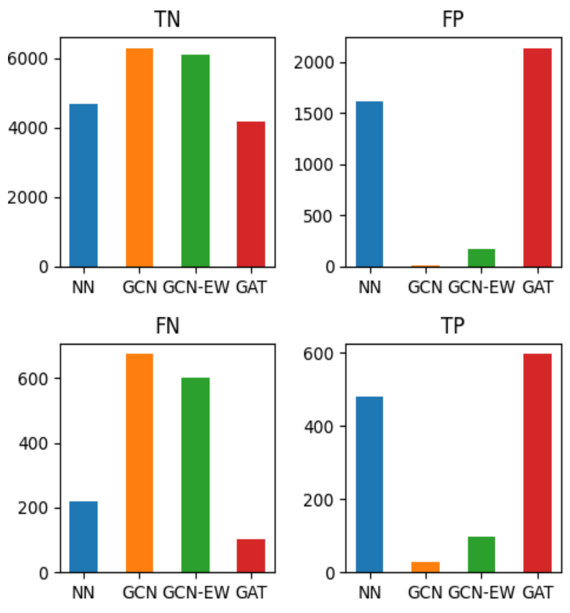

Fig. 5. *
(a) Confusion Matrices for Dataset 1

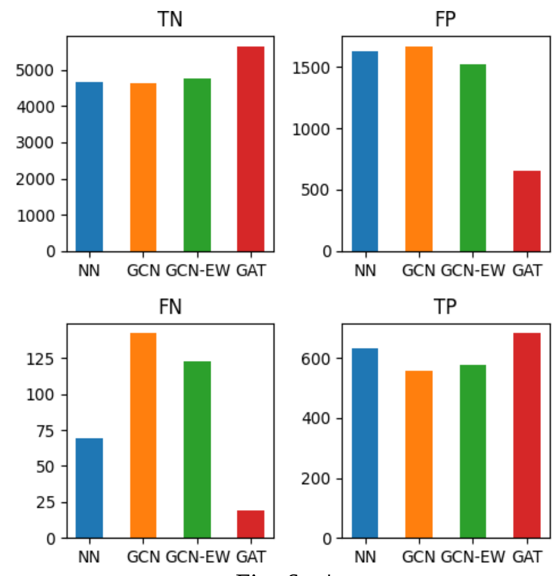

Fig. 6. *

(b) Confusion Matrices for Dataset 2

Fig. 7. Confusion Matrices for Dataset 1 and Dataset 2.

- GAT consistently achieved the highest recall and F1-score across both datasets, making it the most effective model for positive case detection.
- GCN exhibited the highest precision but suffered from lower recall due to its inability to detect all positive instances.
- GCN-EW demonstrated a balance between precision and recall but did not outperform GAT in overall classification capability.
- NN performed competitively but did not match the dis- criminative power of graph-based models.
- Robustness analysis confirmed that GAT remained stable

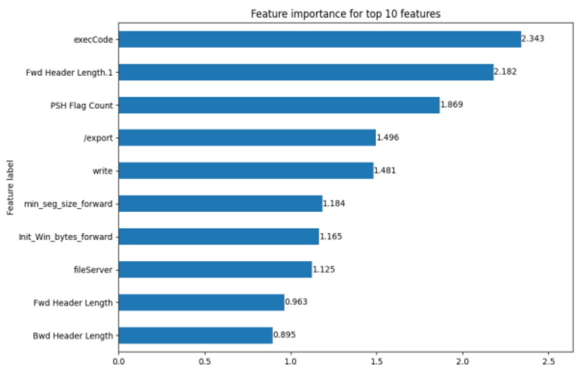

Fig. 8. Top 10 Important Features of Dataset 1.

across varying testing conditions.

These findings suggest that GAT is the most effective model for tasks requiring high recall, while GCN may be preferable in applications where precision is critical. The interpretability analysis further supports GAT's utility by providing insights into decision-making processes.

IX. FUTURE SCOPE

The field of graph-based intrusion detection is rapidly evolving, with numerous opportunities to enhance the capabilities of Graph Neural Network-based Intrusion Detection Systems (GNN-IDS). Future research can focus on refining graph construction techniques and feature engineering by leveraging Temporal Graph Neural Networks (TGNNs). These models can capture long-term attack patterns that develop over time, improving the ability to detect persistent threats. Additionally, integrating diverse data sources, such as *firewall logs, system event logs, and cloud traffic data*, can improve attack correlation and threat detection accuracy.

A. Real-Time Threat Intelligence Integration

Future models can dynamically update with threat intelligence feeds, enabling them to recognize emerging attack patterns. Furthermore, federated learning can facilitate collaboration between multiple IDS deployments while preserving data privacy, ensuring efficient threat detection across distributed networks.

B. Scalable and Distributed GNN Models

Scalability remains a key challenge for GNN-based intrusion detection, as traditional architectures require significant computational resources. To address this, future research should explore the development of lightweight, edge-friendly GNN models capable of operating efficiently in real-time environments. This advancement would support deployment in *cloud security frameworks and IoT networks*, where com- putational efficiency is a major concern.

C. Enhancing Adversarial Robustness

Attackers can manipulate graph structures to evade detection, presenting a significant challenge for GNN-based security systems. To counteract this, future studies should focus on adversarial training techniques to fortify GNN models against such evasion strategies. Developing methods to detect and mitigate *graph adversarial attacks* will ensure that intrusion detection remains effective even against sophisticated cyber threats.

D. Explainability and Interpretability Enhancements

The interpretability of deep learning models, including GNNs, is a crucial concern in cybersecurity. Many existing models function as black-box systems, making it difficult for security professionals to understand the rationale behind threat detections. Incorporating Explainable AI (XAI) techniques can improve transparency by providing clear insights into decision-making processes. Additionally, visualization tools that highlight critical nodes, anomaly scores, and network paths can aid cybersecurity experts in responding to threats more effectively.

E. Automated Response Mechanisms

Integrating Intrusion Prevention Systems (IPS) with GNN-based intrusion detection can enable real-time threat mitigation. This includes *automatically quarantining compromised nodes* and deploying countermeasures against active at- tacks. By leveraging AI-driven threat response, organizations can significantly reduce human intervention, leading to faster and more efficient incident mitigation.

These advancements will play a crucial role in shaping the future of intrusion detection, making systems more adaptive, scalable, and resistant to emerging cyber threats.

X. CONCLUSION

Traditional IDS techniques face challenges like high false positives, limited scalability, and vulnerability to evolving threats. The Graph Neural Network-based Intrusion Detection System (GNN-IDS) addresses these issues by modeling net- work interactions as graphs, enhancing contextual awareness, detection accuracy, and adversarial resilience.

Key advantages include scalability, attack detection efficiency, and forensic interpretability. Future research should focus on federated learning, self-supervised learning, and real- time adaptability to improve IDS effectiveness. GNN-IDS represents a significant step toward intelligent, scalable, and resilient cybersecurity solutions.

REFERENCES

[1] D. E. Denning, ”An Intrusion-Detection Model,” IEEE Transactions on Software Engineering, vol. 13, no. 2, pp. 222–232, Feb. 1987.

[2] M. Roesch, ”Snort - Lightweight Intrusion Detection for Networks,” in Proc. USENIX LISA, 1999.

[3] R. P. Lippmann et al., ”Evaluating Intrusion Detection Systems: The 1998 DARPA Off-line Intrusion Detection Evaluation,” in DARPA IDS Evaluation Program, 2000.

[4] M. Tavallaee, E. Bagheri, W. Lu, and A. Ghorbani, ”A Detailed Analysis of the KDD Cup 99 Dataset,” in IEEE Symposium on Computational Intelligence for Security and Defense Applications, 2009, pp. 1–6.

[5] M. H. Bhuyan, D. K. Bhattacharyya, and J. K. Kalita, ”Network Anomaly Detection: Methods, Systems, and Tools,” IEEE Communi- cations Surveys Tutorials, vol. 16, no. 1, pp. 303–336, 2014.

[6] R. Vinayakumar, K. Soman, and P. Poornachandran, ”Deep Learning Approach for Intelligent Intrusion Detection System,” IEEE Access, vol. 7, pp. 41525–41550, 2019.

[7] J. Liu et al., ”Graph-Based Intrusion Detection System for Large-Scale Networks,” IEEE Transactions on Dependable and Secure Computing, vol. 18, no. 6, pp. 2341–2355, 2020.

[8] D. Zou et al., ”TDGIA: Effective Injection Attacks on Graph Neural Networks,” in Advances in Neural Information Processing Systems (NeurIPS), 2021.

[9] H. Jmal, I. Lagraa, and M. H. A. Hijazi, ”SPGNN-API: A Transferable Graph Neural Network for Attack Paths Identification,” arXiv preprint arXiv:2305.19487, 2023.

[10] B. Van Langendonck, J. P. Menaud, and L. Meunier, ”PPT-GNN: Pretrained Spatio-Temporal Graph Neural Networks for Intrusion De- tection,” arXiv preprint arXiv:2401.06789, 2024.

[11] Bahadir Candan. 2023. Top 5 critical infrastructure cyberattacks.https://www.anapaya.net/blog/top-5-critical-infrastructure- cyberattacks.

[12] Wai Weng Lo, Siamak Layeghy, Mohanad Sarhan, Marcus Gallagher, and Marius Portmann. 2022. E-graphsage: A graph neural network based intrusion detection system for iot. In NOMS 2022-2022 IEEE/IFIP Network Operations and Management Symposium.

IEEE, 1–9.

[13] Ankit Thakkar and Ritika Lohiya. 2022. A survey on intrusion detection system:feature selection, model, performance measures, application perspective, challenges, and future research directions. Artificial Intelligence Review 55, 1 (2022), 453–563.

[14] Benmalek, Mourad, and Kamel-Dine Haouam. ”Advancing Network Intrusion Detection Systems With Machine Learning Techniques.” Advances in Artificial Intelligence and Machine Learning, 2024.

[15] Xu, Renjie, et al. ”Applying Self-supervised Learning to Network Intrusion Detection for Network Flows With Graph Neural Network.” Computer Networks, 2024.

[16] Jmila, Houda, and Mohamed Ibn Khedher. ”Adversarial Machine Learn- ing for Network Intrusion Detection: a Comparative Study.” Computer Networks, 2022.

[17] Sun, Zhenlu, Andre´ M. H. Teixeira, and Salman Toor. ”GNN-IDS: Graph Neural Network-Based Intrusion Detection System.” Proceedings of the 19th International Conference on Availability, Reliability and Security (ARES), 2024.

[18]] Zhitao Ying, Dylan Bourgeois, Jiaxuan You, Marinka Zitnik, and Jure Leskovec.2019. Gnnexplainer: Generating explanations for graph neural networks. Advances in neural information processing systems 32 (2019).

[19] Friji, H., Olivereau, A., Sarkiss, M. (2023). Efficient Network Represen- tation for GNN-Based Intrusion Detection. In International Conference on Security and Privacy in Communication Systems (pp. 532–554). Springer.

[20] Pujol-Perich, D., Sua´rez-Varela, J., Cabellos-Aparicio, A., Barlet- Ros, P. (2022). Unveiling the potential of graph neural networks for robust intrusion detection. ACM SIGMETRICS Performance Evaluation Review, 49(4), 111–117.

[21] Deng, A., Hooi, B. (2021). Graph neural network-based anomaly detection in multivariate time series. In Proceedings of the AAAI Conference on Artificial Intelligence (Vol. 35, No. 5, pp. 4027–4035).

[22] Lo, W. W., Layeghy, S., Sarhan, M., Gallagher, M., Portmann, M. (2022). E-graphsage: A graph neural network based intrusion detection system for iot. In NOMS 2022-2022 IEEE/IFIP Network Operations and Management Symposium (pp. 1–9). IEEE.

[23] ”GNN-NIDS: A Graph Neural Network-Based Network Intrusion Detec- tion System,” *International Conference on Cybersecurity and AI,* 2022. Available: DOI*P LACEHOLDER*.

[24] Hwan Kim, Byung S. Lee, Won-Yong Shin, and Sungsu Lim. 2022. Graph Anomaly Detection With Graph Neural Networks: Current Status and Challenges. IEEE Access 10 (2022), 111820–111829. https://api.semanticscholar.org/CorpusID:252595683

[25] Z. Ahmad, A. Shahid Khan, C. Wai Shiang, J. Abdullah, F. Ahmad, Network intrusion detection system: A systematic study of machine learning and deep learning approaches, Transactions on Emerging Telecommunications Technologies 32 (2021) e4150.

[26] B. Biggio, F. Roli, Wild patterns: Ten years after the rise of adversarial machine learning, in: Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, 2018, pp. 2154–2156.

[27] CapTech, “Emerging Threats to Critical Infrastructure: AI Driven Cybersecurity Trends for 2025,” *CapTech Trends*, Jan. 2, 2025.