Heterogeneous GNN with Express Edges for Intrusion Detection in Cyber-Physical Systems

Hongwei Li
Department of Electrical and Computer Engineering
Villanova University
Email: hli8@villanova.edu

Danai Chasaki
Department of Electrical and Computer Engineering
Villanova University
Email: danai.chasaki@villanova.edu

***Abstract*—With the ever-increasing population of Internet-of-Things (IoT) and Cyber-Physical systems (CPS), cyber attacks can result in significantly more severe consequences. In this paper, we introduce a novel data modeling technique using a heterogeneous graph with Express Edges to enhance the attack detection capabilities of machine learning models. Additionally, we present the first-of-its-kind performance benchmark of representative heterogeneous graph neural network (HGNN) algorithm variants using multiple network intrusion detection system (NIDS) datasets. Our primary aim is to assist CPS defenders in achieving optimal attack detection efficacy against cyber threats.**

***Index Terms*—Cyber-Physical Systems, machine learning, intrusion detection, graph, neural networks, GNN.**

I. Introduction

Cyber-Physical systems (CPS) are expected to play a pivotal role in enabling Industry 4.0 [1]. This reliance on CPS was demonstrated in a recent survey conducted by the authors using the Shodan platform (www.shodan.io). The survey revealed the presence of over 300,000 MODBUS devices (a CPS communication protocol [2]) accessible on the public internet. Safeguarding these increasingly critical assets becomes that much more essential, as the consequences are not limited to mere inconvenience, but can be potentially fatal. This concern is exemplified by incidents like the 2017 Triton malware attack [3] on an oil and gas facilities in Saudi Arabia and the 2021 cyber attack on a North American gas pipeline company [4].

To protect CPS against cyber attacks, Machine Learning (ML) has been proven to be an effective detection tool [5], [6], [7], [8], even for covert attacks designed to evade typical network intrusion detection systems (NIDS). However, the conventional machine learning-based approach not only heavily relies on meticulous feature engineering but also demands deep domain expertise from model designers. Additionally, it is time-consuming due to the iterative nature of the development process.

Another, potentially more fundamental, limitation of traditional machine learning-based NIDS that employs classical ML and Deep Learning (DL) algorithms is the isolation of each packet or network flow from the broader context of the CPS. This shortcoming is particularly evident when dealing with multi-flow formats of attacks, shown in Figure 1, such as those observed in the reconnaissance, Distributed Denial-of-Service (DDoS), and lateral movement stages described in the Mitre ATT&CK [9] framework.

A new breed of deep learning machine learning algorithms, known as Graph Neural Networks (GNN) [10], holds the potential to streamline the feature engineering process. These GNN algorithms have shown promising results that can combine the underlying system topology (graph relationship structure) and other pertinent attributes of CPS components and their interactions, including multi-flow attack patterns [11], [12]. In these GNN-based NIDS systems, network endpoints such as servers or CPS devices are modeled as graph nodes; while network communications, such as packets or flows are modeled as graph edges [13]. Some studies transform the graph into a line graph during data processing, where nodes represent original edges and edges represent original nodes [14], [15]. However, the overall graph is generally treated as a homogeneous graph, where all nodes are required to be of the same type.

In this research, we aim to explore the advantages of an innovative graph modeling approach where a CPS system is modeled as a heterogeneous graph, accommodating multiple types of nodes. This approach allows for diverse source nodes, destination nodes, and network flow nodes. Such an approach empowers CPS defenders with the flexibility to model various device types and relationship types. For example, in a typical CPS system, Master Terminal Units (MTUs) and Remote Terminal Units (RTUs) undertake distinct communication roles, serving as either the source or destination of a network flow. If an RTU, which typically acts as a destination for network flows, suddenly becomes a source of numerous network flows, this anomalous pattern often serves as an indicator of potential intrusion. Moreover, this novel graph data modeling technique is particularly well-suited for representing multi-flow cyber attacks with specific sub-graph typologies, as depicted in Figure 1.

Beyond the innovative graph modeling technique for CPS, we also aim to address a gap in the existing body of published NIDS literature concerning the benefits of different variations of Heterogeneous Graph Neural Networks (HGNN). These variations include neighbor attention, relationship attention, and meta-path attention, which can enhance HGNN-based

979-8-3503-7099-7/24/$31.00 ©2024 IEEE

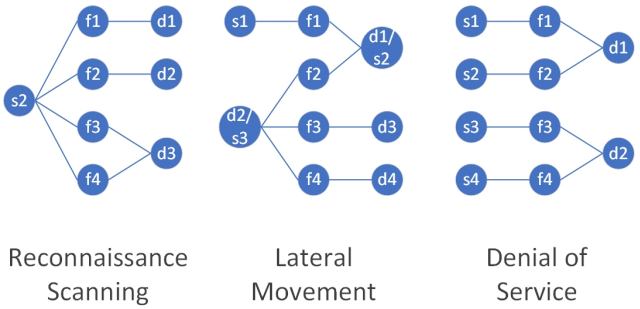

Fig. 1: Sub-graphs illustrating examples of multi-flow attacks. Nodes labeled s_n represent source devices, nodes labeled d_n represent destination devices, and nodes labeled f_n represent different network flows.

NIDS applications. We aim to present the first published performance benchmark of HGNN algorithm variants, with the goal of providing practical guidance to real-world CPS defenders regarding algorithm selection and data processing techniques.

The remainder of this paper is organized as follows: Section II delves into related work. In Section III, we detail the proposed methodology and GNN architecture. Section IV describes the experimental design. Section V presents the evaluation results of the proposed techniques. Finally, Section VI summarizes and concludes the paper.

II. Related Work

The seminal work on Graph Convolutional Networks (GCN) by Kipf and Welling [10] marked a significant advancement in the field of Graph Neural Network (GNN) research. In their work, the authors introduced the concept of applying spectral convolution to non-Euclidean graph data, drawing inspiration from Convolutional Neural Networks (CNNs).

To provide a more general understanding of GNN, a neural message passing framework is employed to exchange vector messages among nodes. These messages are then updated using neural networks. For an input graph $G = (V, E)$, where V represents the vertices or nodes, and E represent the edges, along with a set of node features $X \in \mathbb{R}^{d*|V|}$, the message passing operation from hidden message h_u^k at each layer k for node u to the message at the next layer h_u^{k+1} can be expressed in Equations (1) and (2):

$$h_u^{k+1} = \mathbf{UPDATE}^k(h_u^k, \mathbf{AGG}^k(h_v^k, v \in N(u))), \quad (1)$$

$$= \mathbf{UPDATE}^k(h_u^k, \mathbf{message}_{N(u)}^k), \quad (2)$$

where **UPDATE** and **AGG** (AGGREGATION) are arbitrary differentiable functions used in the neural network. The $\mathbf{message}_{N(u)}$ represents the messages aggregated from neighboring nodes of u.

For the GCN algorithm as described by Kipf and Welling, symmetric normalization is applied in the aggregation function with self-loops, as shown in Equation (3):

$$h_u^{k+1} = \sigma(\mathbf{W}^k \sum_{v \in N(u) \bigcup \{u\}} \frac{h_v^k}{\sqrt{|N(u)||N(v)|}}). \quad (3)$$

To manage memory and facilitate training for large graphs, Hamilton et al. [16] introduced the GraphSAGE algorithm with node neighbor sub-sampling, as shown in Equation (4). Often, the UPDATE function in Equation (2) is substituted with a concatenation component as demonstrated in Equation (5).

$$N_s(u) = \mathbf{SAMPLE}(N(u), SampleSize), \quad (4)$$

$$h_u^{k+1} = \sigma(W^k \cdot \mathbf{CONCAT}(h_u^k, \mathbf{message}_{N_s(u)}^k)). \quad (5)$$

Lo et al. [13] extended the GraphSAGE algorithm to include edge features for classifying network source and destination devices as nodes, and network flows as edges with flow features in a homogeneous graph, as illustrated in Figure 2a. Source IP addresses are randomly mapped to one of 16 addresses. With this E-GraphSAGE modification, Lo et al. reported generally improved performance compared to ensemble tree-based ML models.

Chang and Branco [14] further improved the E-GraphSAGE model and demonstrated the benefits of applying attention to GNN for enhanced NIDS performance. In order to leverage existing body of techniques in node classification, this work transformed the graph into a homogeneous line-graph counterpart, where nodes correspond to original edges and vice versa, as depicted in Figure 2b. Source IP addresses are also randomly mapped to one of 16 addresses. Friji et al. [15] also applied the same line-graph technique along with graph attention mechanism. When it comes to model performance validation, Friji et al. highlighted a crucial concern regarding potential target leakage resulting from randomized splits of train-test samples in most published NIDS dataset performance tests. The primary reason for this issue is the limited size of the testbed in most CPS NIDS datasets, where only a small number of source IP addresses initiate simulated cyber attacks. However, as shown in Figure 3, some simulated attacks in the ToN-IoT dataset include only 2 or 3 source IP addresses, making the IP Address based train-test split solution proposed by Friji et al. unsuitable for all NIDS datasets.

Pujol-Perich et al. [17] conducted experiments to showcase the robustness of GNN against simulated adversarial attacks, such as variations in packet size or inter-arrival times. They demonstrated that by leveraging relationship information within the graph structure, their implementation of the GNN model exhibited more stable performance compared to ensemble tree-based and MLP models. For graph data modeling, Pujol-Perich et al. represented both network devices and network flows as nodes in a heterogeneous graph, shown in Figure 2c.

Many previous studies have focused on utilizing homogeneous graphs for data modeling, resulting in the restriction of all nodes to a single type. However, within real-life enterprise

IT networks, a diverse range of device types or roles is present, encompassing servers, user computers, mobile devices, IoT devices, and CPS devices. Communication patterns for CPS devices, especially those deployed in critical infrastructures such as power or transportation systems, exhibit even greater diversity. This diversity underscores the need for a heterogeneous graph approach that can comprehensively capture the intricacies of real-world networks, thereby forming a more robust foundation for GNN graph representation learning. Moreover, this innovative graph data modeling technique is well-suited to depicting multi-flow cyber attacks in distinct sub-graph typologies, as illustrated in Figure 1.

In addition, the concerns raised by Friji et al. [15] and Engelen et al. [18] regarding model evaluation highlight the potential for target leakage arising from randomized data and train-test sample splits. While splitting based on time is a common strategy to mitigate this issue in real-life NIDS data, it may not effectively address the challenge posed by simulated datasets. These datasets frequently lack a normal distribution of traffic types over time due to bursty simulation behavior or may divide a single actual network flow into multiple flow records. To address these limitations, we propose a modified temporal train-test split of the dataset coupled with the diversification of source IP addresses. This approach aims to yield evaluation results that better reflect real-world scenarios, as discussed in Section V.

III. Proposed Methodology

A. Hetergeneous Graph Data Modeling with Express Edges

To accurately model the complexity of actual CPS networks, which encompass various device types with distinct network behavior attributes, we propose a heterogeneous graph data model. This data model is designed for its extensibility to accommodate the evolving landscape of CPS technology and applications.

A heterogeneous graph, denoted as $G = (V, E)$, consists of a set of vertices or nodes V and a set of edges E. Each node $v \in V$ is associated with its mapping function $\phi(v) : V \rightarrow A$, where A denotes the node type set. Similarly, each edge $e \in E$ is associated with it's mapping function $\phi(e) : E \rightarrow R$, where R denotes the edge type set. When $|A| = 1$ and $|R| = 1$, the graph degenerate into a homogeneous graph. An example heterogeneous graph is an academic bibliographic graph, where node types could be defined as paper, author, venue, and term, while edge types could be write, contain, publish, and so on.

In the context of CPS networks, node types can correspond to CPS device roles, such as MTU, RTU, HMI, Engineering Station, and more. In this study, since the roles of the CPS devices are not known to the authors, Source IP address (src) and Destination (dst) IP address are used as a proxy for what might be represented in real-life scenarios. To facilitate the representation of multi-flow attack typologies illustrated in Figure 1, we also introduce Netflow flows to the node type set, representing each network flow as a node. Undirected edges are employed to connect src/dst nodes with flow nodes, as depicted in Figure 2d.

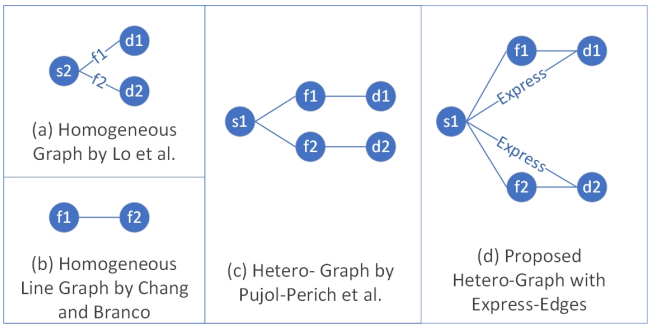

Fig. 2: Different graph data modeling techniques: (a) homogeneous graph used by Lo et al. in [13]; (b) homogeneous line-graph used by Chang and Branco [14]; (c) heterogeneous graph used by Pujol-Perich et al. [17]; (d) Proposed heterogeneous graph with Express Edges.

To expedite message passing within the Heterogeneous Graph Neural Network (HGNN), we introduce **Express Edges** between Source (src) and Destination (dst) nodes, as depicted in Figure 2d. The forthcoming results in Section V will showcase the advantages of incorporating **Express Edges** into the graph.

B. Relational Graph Convolutional Networks (RGCN) and Relational GraphSage (Rsage)

Schlichtkrull et al. [19] extended the work of Kipf and Welling [10] to heterogeneous graphs by accounting for edge type relationship set R in the GCN Equation (3) as shown in Equation (6):

$$h_u^{k+1} = \sigma(\sum_{r \in R} \sum_{v \in N_u^r} \frac{1}{C_{u,r}} W_r^k h_v^k + W_0^k h_u^k), \tag{6}$$

where N_u^r denotes the set of neighbor indices of node u under relation $r \in R$. $c_{u,r}$ is a normalization constant that can either be learned or chosen in advance (such as $c_{u,r} = |N_u^r|$).

Similarly, the GraphSAGE [16] algorithm was extended to heterogeneous graphs (Rsage) by accounting for edge type relationship in the GraphSage Equation (5).

C. Relational Graph Attention Networks (RGAT, and HAN)

Graph Attention Networks (GAT) [20] introduced an attention mechanism initially applied to enhance Natural Language Processing capabilities [21]. By learning individual attention weights for each neighbor of a given node, GAT aims to achieve improved information aggregation. The attention weight α_{uv} is defined as:

$$\alpha_{uv} = \frac{\exp(\sigma(a^T[Wh_u, Wh_v]))}{\sum_{k \in \mathcal{N}_u} \exp(\sigma(a^T[Wh_u, Wh_k]))}, \tag{7}$$

Subsequently, the node representation h'_u is obtained by combining neighboring node embeddings using the attention weights:

$$h'_u = \sigma(\sum_{v \in N(u)} \alpha_{uv} W h_v). \quad (8)$$

Finally, similar to Equation (6) used for RGCN, the introduction of edge type relations leads to the development of Relationship GAT (RGAT).

Wang et al. [22] published Heterogeneous Graph Attention Network (HAN) algorithm by adding another layer of attention that learns the different attention weights amongst different meta-paths. Meta-paths are human curated information of meaningful path vectors within the graph, such as "Paper-Author-Paper", or "Paper-Term-Paper" in an academic bibliographic graph.

D. Modified Temporal Data Split and Diversification of Source IP Addresses

In real-life NIDS datasets, a common approach to prevent target leakage issues is to perform train-test splitting based on time. This involves reserving the most recent data, such as the last 3 weeks, for testing, and utilizing earlier data for training and validation purposes. However, to address the unique characteristics of simulated NIDS datasets and to mitigate potential target leakage problems, we propose a modified temporal split strategy that is particularly suitable for most simulated NIDS datasets.

1) Strategy 1: Temporal Data Split: In order to address the bursty nature of simulation data, especially in attack traffic scenarios typical of simulation datasets, we suggest a data splitting approach that considers both attack types and the temporal dimension of the simulation. For instance, in the ToN-IoT dataset, we divide the last 30% of flows for each traffic type during each simulation day. For example, for XSS attacks, we sort the flows by timestamp, using source IP address and source port number as secondary sorting criteria to resolve ties. The final 30% of flows are then reserved for testing. For datasets lacking explicit timestamps, the chronological order of the data can be used in conjunction with sub-experiment types.

2) Strategy 2: Diversification of Source IP Address: To mitigate target leakage issues arising from a limited number of source devices during attack and normal traffic simulations, we propose concatenating the original source IP addresses with the timestamp of the flow (including Year, Month, Day, Hour, Minute, and Second). This approach creates the illusion of a new set of source devices every second. Since raw IP addresses are typically not used as features in GNN models, this strategy prevents the models from making inferences based solely on source IP address relationships. When combined with the modified temporal train-test sample splitting, this strategy helps minimize target leakage issues. This fine-grained time resolution is particularly crucial for the ToN-IoT dataset, where the shortest burst of XSS attack simulation lasted for 93 seconds. For other simulated datasets, the time resolution can be adjusted to achieve the desired diversification effect.

For datasets lacking timestamps, such as NF-BoT-IoT [23], concatenating the source IP addresses with the source port can serve as a suitable workaround, given that most new TCP flows tend to choose a different port number at the source.

An important advantage of the proposed Diversification and Temporal Data Split is the deterministic nature of the resulting data splits. This feature enables easy reproduction across different research teams, facilitating direct comparison of study results.

IV. Evaluation Data and Experimental Design

A. CPS Evaluation Data: ToN-IoT dataset

The ToN-IoT [24] network dataset, frequently referenced in related GNN-NIDS studies, is chosen for evaluating the proposed CPS NIDS methodology and for facilitating comparative analyses. This relatively recent intrusion detection dataset models a substantial number of CPS and IoT devices, at the Cyber Range and IoT Labs at UNSW Canberra, Australia. Alongside real devices like smartphones and smart TVs, the dataset includes simulated CPS devices such as Fridges, GPS, and Thermostat sensors. The dataset encompasses various types of cyber-attacks, including scanning, DoS, DDoS, ransomware, backdoor, data injection, cross-site scripting (XSS), password cracking, and Man-in-The-Middle (MITM) attacks. In the Train_Test version of the dataset, there are 161,043 Attack flows compared to 300,000 Normal flows. A detailed dataset analysis is presented in Figure 3.

B. Additional Evaluation Data

In order to assess the broader applicability of the proposed methodology, we extend our analysis to include additional datasets widely utilized in state-of-the-art NIDS studies. These datasets consist of CIC-IDS2017 [25], CIC-Darknet [26], and NF-BoT-IoT [27]. An overview of the key statistics for these datasets is summarized in Table I.

TABLE I: Statistics of Dataset Used in This Study

Dataset	Normal Flows	Attack Flows	Normal:Attack Ratio
ToN-IoT	300,000	161,043	65.1% : 34.9%
CIC-IDS2017	1,657,693	443,121	78.9% : 21.9%
CIC-Darknet	117,219	24,311	82.8% : 17.2%
NF-BoT-IoT	13,859	586,241	2.3% : 97.7%

Most datasets exhibit an expected class imbalance, reflecting real-world scenarios where the majority of data points belong to the "Normal" class. Notably, NF-BoT-IoT displays a pronounced skew towards the "Attack" class, with only 2.3% of data representing the "Normal" class.

C. Performance Metrics and Hyper-parameters

Given the imbalanced nature of the dataset, the F1 Score is chosen as the performance metric for comparisons. The F1 Score provides a balanced view between predicted and actual positives, independent of potentially high numbers of true

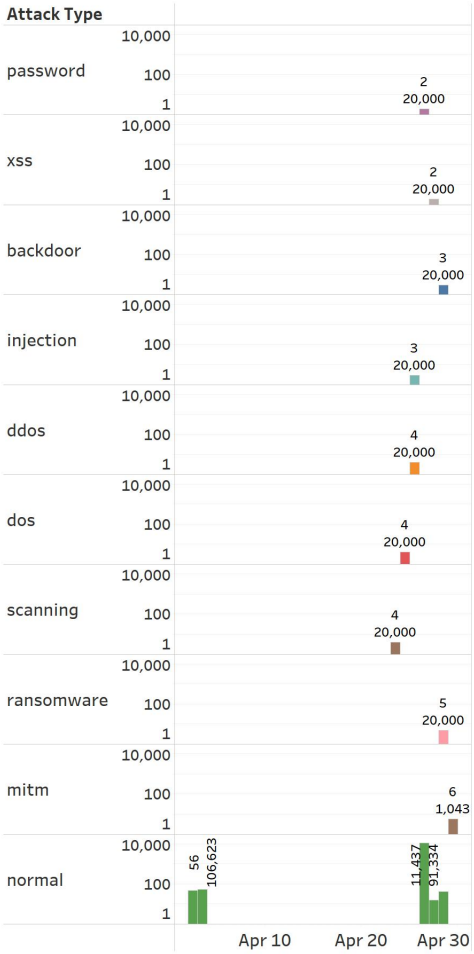

Fig. 3: Daily distribution of Source IP Addresses (unique addresses) in the ToN-IoT dataset. The first number above each bar represents the count of distinct source IP Addresses, while the second number represents the total number of network flows observed on that day.

negatives (TN) or normal samples. The F1 Score is calculated using the formula:

$$F1Score = \frac{2 \times TP}{2 \times TP + FP + FN} \tag{9}$$

To ensure fair comparison of model results, the hyper-parameters for most GNN algorithms in this study are set as follows: Number of GNN layers is 2; mini-batch size is 2048; hidden dimension is 64 for ToN-Iot and NF-BoT-Iot is 64, and 128 for CIC-IDS2017 and CIC-Darknet to accommodate more features in these two datasets; the models are trained for 3 epochs each with dropout ratio of 0.3 and a learning rate of 5e-3.

For computational time evaluation, all experiments were conducted on a computer equipped with an Intel i7-11700 CPU with 16 cores, 64 GB memory, and an NVIDIA GeForce RTX3070 GPU with 8GB video memory. Software libraries mainly consist of PyTorch, DGL, and OpenHGNN [28].

V. Results and Discussions

A. Comparison of Model Results with State-of-the-Art Studies

Binary classification results are employed in this study to enable comparisons with other published State-of-the-Art (SOTA) studies on ML-based NIDS. Table II presents results from publications that utilized data splitting methods aimed at mitigating the target leakage effects from random data splitting.

Among methods with similar experimental setups, the RSAGE and RGCN algorithms utilizing the **Express Edge** technique achieved the most promising results.

Figure 4 illustrates the computational time for different GNN algorithms. Notably, RGCN and RSAGE exhibit favorable performance in terms of both prediction accuracy and computational efficiency.

TABLE II: Comparison of Performance with State-of-the-Art Studies

ML Algorithm	Study	F1-Score
MLP NN	Friji et al.[15]	0.33438
XG-Boost	Friji et al.[15]	0.4807
E-graphsage	Lo et al. [13] Friji et al. [15]	0.88
Dual-Relation GNN	Friji et al. [15] Pujol-Perich et al. [17]	0.902
GNN with Residuals	Friji et al.[15]	0.937
RGCN with Express Edges	This study	**0.9735**
RSAGE with Express Edges	This study	**0.9778**
RGAT without Express Edges	This study	0.9423
HAN without Express Edges	This study	0.9688

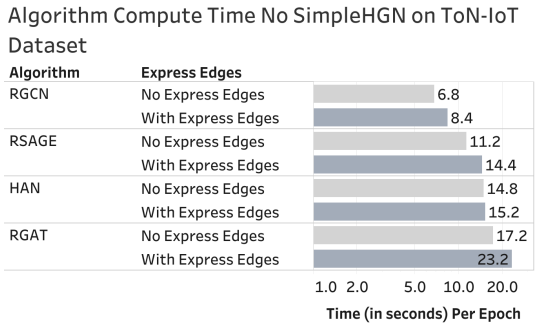

Fig. 4: Compute time study in seconds per epoch.

B. Performance Benchmark across Multiple Datasets

To assess the broader applicability of the proposed **Express Edge** methodology, systematic experimental results for three additional datasets are presented in Table III. Each performance data point is an average of five independent runs ($n = 5$), except for the Rsage algorithm, where ($n = 10$) compensates for the inherent randomness in the neighbor sampling process.

We observe that models with **Express Edges** exhibit superior performance for most datasets, except for the NF-BoT-IoT dataset, which is heavily skewed toward attack data.

TABLE III: Performance Benchmark across Multiple Datasets

Dateset	Algorithm	Graphs with Express Edges	Graphs without Express Edges
ToN-IoT	RSAGE	**0.9778**	0.9739
	RGCN	0.9735	0.9732
	RGAT	0.9312	0.9423
	HAN	0.9687	0.9688
CIC-IDS2017	RSAGE	**0.9871**	0.9865
	RGCN	0.8829	0.9411
	RGAT	0.9212	0.9212
	HAN	0.9832	0.9832
CIC-Darknet	RSAGE	0.9031	0.9155
	RGCN	0.9041	0.8832
	RGAT	**0.9186**	0.9132
	HAN	0.9146	0.9123
NF-BOT-IOT	RSAGE	0.6977	0.6943
	RGCN	0.7124	0.7243
	RGAT	0.7309	**0.7440**
	HAN	0.7024	0.7025

Another notable observation from Table III is that while Rsage with **Express Edge** methods produced optimal results for two datasets, RGAT and HAN occasionally outperformed other models in different contexts. Until a consistently superior HGNN algorithm emerges, it is advisable to compare the performance of representative HGNN algorithms to select the most suitable one for the specific problem, similar to the comparative study illustrated in Table III.

VI. Conclusion

In this paper, we introduce a novel approach to intrusion detection in cyber-physical systems, harnessing the power of Heterogeneous Graph Neural Networks with **Express Edges** during the graph construction phase. The incorporation of **Express Edges**, which establish direct connections between communicating network device nodes, accelerates the GNN message passing process and yields superior model performance when compared to other graph modeling techniques from prior studies.

Furthermore, our study stands out as the first NIDS investigation to propose a refined temporal data splitting method and source IP address diversification strategy. These innovations aim to foster more realistic evaluation of model performance results, mitigating potential target leakage issues inherent in conventional random split approaches. By introducing a deterministic approach, we also facilitate comparative analyses across different research teams.

Through the utilization of the novel **Express Edge** methodology and an improved evaluation process, this work establishes a pioneering benchmark by systematically evaluating a range of representative HGNN algorithms across multiple NIDS datasets. The results underscore the necessity of employing a rigorous comparative study methodology to attain optimal attack detection outcomes. It becomes evident that there is currently no universally superior HGNN algorithm that excels across all tasks and NIDS datasets. Thus, our study sheds light on the importance of tailoring algorithmic choices to specific problem contexts for achieving optimal cyber attack detection performance.

In future studies, we plan to review performance of additional published HGNN algorithms by leveraging the proposed evaluation methodology and aim to improve HGNN algorithm performance based on insights gained through performance bench-marking of various HGNN algorithms.

References

[1] N. Jazdi, "Cyber physical systems in the context of Industry 4.0," in *2014 IEEE International Conference on Automation, Quality and Testing, Robotics*, May 2014, pp. 1–4.

[2] Modbus.org, "Modbus Application Protocol V1.1b3," 2012. [Online]. Available: http://www.modbus.org/docs/

[3] S. Miller, N. Brubaker, D. K. Zafra, and D. Caban, "TRITON Actor TTP Profile, Custom Attack Tools, Detections, and ATT&CK Mapping," 2019. [Online]. Available: https://www.fireeye.com/blog/threat-research/2019/04/triton-actor-ttp-profile-custom-attack-tools-detections.html

[4] J. R. Reeder and T. Hall, "Cybersecurity's Pearl Harbor Moment: Lessons Learned from the Colonial Pipeline Ransomware Attack," *The Cyber Defense Review*, vol. 6, no. 3, pp. 15–40, 2021, publisher: Army Cyber Institute. [Online]. Available: https://www.jstor.org/stable/48631153

[5] M. H. Bhuyan, D. K. Bhattacharyya, and J. K. Kalita, "Network Anomaly Detection: Methods, Systems and Tools," *IEEE Communications Surveys Tutorials*, vol. 16, no. 1, pp. 303–336, 2014, conference Name: IEEE Communications Surveys Tutorials.

[6] M. A. Al-Garadi, A. Mohamed, A. Al-Ali, X. Du, I. Ali, and M. Guizani, "A Survey of Machine and Deep Learning Methods for Internet of Things (IoT) Security," *IEEE Communications Surveys Tutorials*, pp. 1–1, 2020.

[7] D. Chou and M. Jiang, "A Survey on Data-driven Network Intrusion Detection," *ACM Computing Surveys*, vol. 54, no. 9, pp. 182:1–182:36, Oct. 2021. [Online]. Available: https://dl.acm.org/doi/10.1145/3472753

[8] H. Li and D. Chasaki, "Network-Based Machine Learning Detection of Covert Channel Attacks on Cyber-Physical Systems," in *2022 IEEE 20th International Conference on Industrial Informatics (INDIN)*, Jul. 2022, pp. 195–201.

[9] "MITRE ATT&CK®." [Online]. Available: https://attack.mitre.org/

[10] T. N. Kipf and M. Welling, "Semi-Supervised Classification with Graph Convolutional Networks," Feb. 2017, arXiv:1609.02907 [cs, stat]. [Online]. Available: http://arxiv.org/abs/1609.02907

[11] G. Dong, M. Tang, Z. Wang, J. Gao, S. Guo, L. Cai, R. Gutierrez, B. Campbel, L. E. Barnes, and M. Boukhechba, "Graph Neural Networks in IoT: A Survey," *ACM Transactions on Sensor Networks*, vol. 19, no. 2, pp. 47:1–47:50, Apr. 2023. [Online]. Available: https://dl.acm.org/doi/10.1145/3565973

[12] T. Bilot, N. E. Madhoun, K. A. Agha, and A. Zouaoui, "Graph Neural Networks for Intrusion Detection: A Survey," *IEEE Access*, vol. 11, pp. 49 114–49 139, 2023, conference Name: IEEE Access.

[13] W. W. Lo, S. Layeghy, M. Sarhan, M. Gallagher, and M. Portmann, "E-GraphSAGE: A Graph Neural Network based Intrusion Detection System," *arXiv:2103.16329 [cs]*, Jul. 2021, arXiv: 2103.16329. [Online]. Available: http://arxiv.org/abs/2103.16329

[14] L. Chang and P. Branco, "Graph-based Solutions with Residuals for Intrusion Detection: the Modified E-GraphSAGE and E-ResGAT Algorithms," Nov. 2021, arXiv:2111.13597 [cs]. [Online]. Available: http://arxiv.org/abs/2111.13597

[15] H. Friji, A. Olivereau, and M. Sarkiss, "Efficient Network Representation for GNN-Based Intrusion Detection," in *Applied Cryptography and Network Security*, ser. Lecture Notes in Computer Science, M. Tibouchi and X. Wang, Eds. Cham: Springer Nature Switzerland, 2023, pp. 532–554.

[16] W. L. Hamilton, R. Ying, and J. Leskovec, "Inductive Representation Learning on Large Graphs," Sep. 2018, arXiv:1706.02216 [cs, stat]. [Online]. Available: http://arxiv.org/abs/1706.02216

[17] D. Pujol-Perich, J. Suárez-Varela, A. Cabellos-Aparicio, and P. Barlet-Ros, "Unveiling the potential of Graph Neural Networks for robust Intrusion Detection," *arXiv:2107.14756 [cs]*, Jul. 2021, arXiv: 2107.14756. [Online]. Available: http://arxiv.org/abs/2107.14756

[18] G. Engelen, V. Rimmer, and W. Joosen, "Troubleshooting an Intrusion Detection Dataset: the CICIDS2017 Case Study," in *2021 IEEE Security and Privacy Workshops (SPW)*, May 2021, pp. 7–12.

[19] M. Schlichtkrull, T. N. Kipf, P. Bloem, R. van den Berg, I. Titov, and M. Welling, "Modeling Relational Data with Graph Convolutional Networks," in *The Semantic Web*, ser. Lecture Notes in Computer Science, A. Gangemi, R. Navigli, M.-E. Vidal, P. Hitzler, R. Troncy, L. Hollink, A. Tordai, and M. Alam, Eds. Cham: Springer International Publishing, 2018, pp. 593–607.

[20] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph Attention Networks," Feb. 2018, arXiv:1710.10903 [cs, stat]. [Online]. Available: http://arxiv.org/abs/1710.10903

[21] D. Bahdanau, K. Cho, and Y. Bengio, "Neural Machine Translation by Jointly Learning to Align and Translate," May 2016, arXiv:1409.0473 [cs, stat]. [Online]. Available: http://arxiv.org/abs/1409.0473

[22] X. Wang, H. Ji, C. Shi, B. Wang, P. Cui, P. Yu, and Y. Ye, "Heterogeneous Graph Attention Network," Jan. 2021, arXiv:1903.07293 [cs]. [Online]. Available: http://arxiv.org/abs/1903.07293

[23] M. Sarhan, S. Layeghy, N. Moustafa, and M. Portmann, "NetFlow Datasets for Machine Learning-based Network Intrusion Detection Systems," *arXiv:2011.09144 [cs]*, vol. 371, pp. 117–135, 2021, arXiv: 2011.09144. [Online]. Available: http://arxiv.org/abs/2011.09144

[24] A. Alsaedi, N. Moustafa, Z. Tari, A. Mahmood, and A. Anwar, "TON_iot Telemetry Dataset: A New Generation Dataset of IoT and IIoT for Data-Driven Intrusion Detection Systems," *IEEE Access*, vol. 8, pp. 165 130–165 150, 2020, conference Name: IEEE Access.

[25] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization." *ICISSp*, vol. 1, pp. 108–116, 2018.

[26] A. Habibi Lashkari, G. Kaur, and A. Rahali, "DIDarknet: A Contemporary Approach to Detect and Characterize the Darknet Traffic using Deep Image Learning," in *Proceedings of the 2020 10th International Conference on Communication and Network Security*, ser. ICCNS '20. New York, NY, USA: Association for Computing Machinery, Mar. 2021, pp. 1–13. [Online]. Available: https://dl.acm.org/doi/10.1145/3442520.3442521

[27] M. Sarhan, S. Layeghy, and M. Portmann, "Towards a Standard Feature Set for Network Intrusion Detection System Datasets," *Mobile Networks and Applications*, vol. 27, no. 1, pp. 357–370, Feb. 2022. [Online]. Available: https://doi.org/10.1007/s11036-021-01843-0

[28] H. Han, T. Zhao, C. Yang, H. Zhang, Y. Liu, X. Wang, and C. Shi, "OpenHGNN: An Open Source Toolkit for Heterogeneous Graph Neural Network," in *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, ser. CIKM '22. New York, NY, USA: Association for Computing Machinery, Oct. 2022, pp. 3993–3997. [Online]. Available: https://dl.acm.org/doi/10.1145/3511808.3557664