Received 26 July 2025, accepted 25 August 2025, date of publication 27 August 2025, date of current version 4 September 2025.

Digital Object Identifier 10.1109/ACCESS.2025.3603403

APPLIED RESEARCH

A Hybrid Graph Neural Network Model for Predicting Cyber Attacks From Heterogeneous and Dynamic Network Data

MÜCAHIT SOYLU[1] **AND RESUL DAS**[2]
[1]Organized Industrial Zone Vocational School Computer Programming, İnonu University, 44280 Malatya, Türkiye
[2]Faculty of Technology, Department of Software Engineering, Firat University, 23119 Elazig, Türkiye
Corresponding author: Mücahit Soylu (mucahit@inonu.edu.tr)

ABSTRACT With valuable data constantly under attack, reactive security measures are no longer sufficient. Predicting cyber threats before they emerge is crucial. Cyberattacks do not occur randomly; they have a systematic underlying pattern. By discovering these patterns, it is possible to predict cyberattacks in advance. Unraveling the mysteries of these evolutionary patterns is quite challenging. Considering the potential of Graph Neural Networks to strengthen cybersecurity defenses, this paper proposes a new hybrid model, DyMHAG (Dynamic Meta-Path Heterogeneous Attention Graph). We propose a novel hybrid GNN model that integrates meta-path-based graph attention networks with the Gated Recurrent Unit (GRU) mechanism for temporal data processing. Our method consists of three layers: node-level attention-based graph embedding, meta-path-level attention-based graph embedding, and evolutionary pattern learning. This model aims to effectively capture complex relational structures and temporal dependencies in heterogeneous and temporally dynamic network data and provide a proactive solution for cyberthreat prediction. Preliminary evaluations indicate that our hybrid model not only improves prediction accuracy but also reduces the false positive rate, providing a more reliable defense against emerging cyber threats.

INDEX TERMS Cyber thread prediction, graph neural networks, heterogeneous graph, meta-path, temporal dynamic networks.

I. INTRODUCTION

The constant threat of cyberattacks looms large in our hyperconnected world. As wireless communication and internet connectivity explode, cybercriminals exploit new vulnerabilities, harming individuals, businesses, and governments [1]. Traditional detection methods struggle to keep pace with the ever-evolving nature of these threats. This necessitates the development of innovative solutions. The rise of artificial intelligence, with its potential for both new opportunities and complex threats, further underscores this critical need [2]. Attackers can easily infiltrate target systems with an increasing number of attack tools and advanced attack techniques. This makes traditional signature-based security protection strategies inadequate.

The associate editor coordinating the review of this manuscript and approving it for publication was Yiqi Liu.

To prevent unknown cyber risks, it has become critical to proactively predict potential threats. For this purpose, various cyber threat prediction methods have been developed for different scenarios. Most of these methods focus on analyzing cyber threat reports, system logs or network traffic to predict cyber threats. One approach that has received a great deal of attention is the application of Graph Neural Networks (GNN). GNN are a type of artificial neural network that can effectively model and analyze complex relationships in graph-structured data [3]. These networks have emerged as a promising approach to improve the effectiveness of defense measures [1] due to their ability to process and learn from heterogeneous cyber threat data [4]. These different characteristics and the relentless evolution of cyber attacks have led cyber defense to adopt modern approaches such as GNN to strengthen defensive measures and break the attack lifecycle.

Data mining has been used to tackle cybersecurity challenges as it can discover actionable patterns from data.

Given that cybersecurity data can be expressed in graphs, GNNs have analyzed cybersecurity data more meaningful and easier. In recent years, variants of GNNs such as Graph Convolution Networks (GCN), Graph Attention Networks (GAT), Graph Recurrent Network (GRN) have shown breakthrough performances in many deep learning tasks. [3]. The first example of GNNs goes back to a long history of neural networks for graphs. In the nineties, Recursive Neural Networks were first used for directional cyclic graphs [5]. Later, Recurrent Neural Networks [6] and Feedforward Neural Networks [7] were introduced to handle loops. However, since the universal idea behind these methods is to build state transition systems on graphs and iterate until convergence, it has limited their extensibility and representativeness. Recent advances in deep neural networks Convolutional Neural Networks (CNN)s have led to the rediscovery of GNNs. CNNs can extract and combine multi-scale local spatial features, enabling them to create highly expressive representations, which has revolutionized almost all areas of machine learning and ushered in a new era of deep learning. This development led to the idea of generalizing CNNs to graphs. However, defining local convolution filters and pooling operators is difficult, which hinders the transformation of Convolutional Neural Networks from Euclidean space to non-Euclidean space. The process of extending deep neural models to non-Euclidean domains, often referred to as geometric deep learning, is an emerging research area. Under this general term, deep learning on graphs has garnered significant interest.

In the field of graph analysis, traditional machine learning approaches often rely on handcrafted features, limiting flexibility and being costly. Following the success of representation learning and word embeddings (SkipGram) [8], DeepWalk [9] is considered the first graph embedding method based on representation learning, which applies the SkipGram [8] model on generated random walks. Similar approaches like Node2vec [10] have also made significant progress. However, these methods have two major drawbacks [11]. Firstly, they lack shared parameters between nodes in the encoder, leading to computational inefficiency as the number of parameters increases linearly with the number of nodes. Secondly, direct embedding methods lack generalization ability, meaning they cannot handle dynamic graphs or generalize to new graphs.

As traditional methods struggle with ever-evolving cyber threats, advancements in GNNs offer a proactive solution. By effectively modeling complex relationships in network data, GNNs, particularly meta-path-based heterogeneous GNNs, can analyze diverse threat information and disrupt attack cycles. Integrating these techniques with deep learning holds immense potential for enhancing cyber defenses. In our study, we propose a hybrid model combining meta-path-based graph attention networks with Gated Recurrent Unit (GRU) mechanism for temporal data processing. This integration aims to leverage the strengths of both techniques in capturing complex relational structures and temporal dependencies. The model's attention mechanism focuses on relevant nodes and edges, while GRU mechanism enhances its ability to capture temporal dynamics. Our approach shows promise in the domain of cyber-threat prediction, offering effective analysis of interconnected data with temporal aspects.

A. PROBLEM STATEMENT

Today, there are many different studies on the detection and prevention of cyber attacks. In particular, there are many studies on artificial intelligence-based attack detection and prediction using classical artificial intelligence models. In recent years, there are studies conducted with new generation artificial intelligence methods such as GNN, CNN, etc. However, when these studies are analyzed, it is a big problem to detect and predict attacks from complex, heterogeneous and large-sized dynamic cyber attack data. There are very limited studies in the literature to solve this problem. Therefore, in order to improve cyber threat detection and mitigation strategies and ultimately strengthen information system security against evolving threats, it is imperative to develop advanced models that can predict and prevent cyber threats before they occur by better understanding the dynamics and complex structure of networks.

B. MAIN CONTRIBUTIONS

In this study, a new hybrid model based on GNNs is proposed for cyber attack detection and prediction using heterogeneous, complex, large-scale, and dynamic cyber attack data. Due to the proven success of node-level attention-based graph embedding and meta-path level attention-based graph embedding, these methods are integrated with the temporal processing mechanism GRU to propose a hybrid cyber threat analysis model. The developed and proposed hybrid model for cyber attack detection and prediction has provided significant and versatile contributions.

- To analyze the complex relational structures in cyber threat data, advanced graph embedding techniques such as node-level attention-based graph embedding and graph embedding based on meta-path level attention were initially employed. This allowed for a thorough investigation of the relational structures within cyber attack data, enabling the precise definition of relationships between nodes in complex network structures and facilitating the identification of threat origins and propagation patterns.
- To analyze and predict evolving cyber threats over time, GRU was employed to learn evolutionary patterns in time-series data. With this method, the evolution of past threats was tracked to predict future threat trends. Consequently, it enables cybersecurity experts to understand the dynamics of threats over time and develop appropriate defense strategies.
- By reducing the number of processing steps in the analysis process of big data, a significant increase in performance has been achieved in predicting cyber attacks.

C. STRUCTURE OF THE PAPER

The first section of the article introduces background information about cyber attack prevention methods that have been used so far, the motivation of this paper, and the study's main contributions. The main concepts used in this paper are described in *the preliminaries section*. *The third section*, literature review, covers the prominent studies in the field of cyber attacks. GNN models which analyze heterogeneous and dynamic heterogeneous data are described in Table 2 as well. The proposed methodology, datasets, data processing, and the proposed model are described in *section four*. In *section five*, experimental results are presented and discussed. Lastly, in *section six*, conclusions regarding the highlights of the study are explained and suggestions for further research are presented.

II. PRELIMINARIES

In this section, we define the concepts of meta-path, meta-path-based neighborhood, heterogeneous graph of cyber threats, and network embedding. These concepts are essential for our model.

Definition 1: (Meta-path): A meta-path is a relationship template that describes the types of connections between two nodes. We can show it as $v_1 \xrightarrow{R_1} v_2 \xrightarrow{R_2} \cdots \xrightarrow{R_l} v_n$, Here, it defines a composite relationship denoted by $R = R_1 \diamond R_2 \diamond \cdots \diamond R_n$ where $\diamond$ represents the composition operation between nodes, v_1 and v_n. Imagine we have computers on a network. Meta-paths, which describe the sequence of steps a cyberattacker might take to infiltrate a system (e.g., attacker exploits a vulnerability in a website, downloads malware onto a user's computer, steals sensitive information), reveal the specific attack pathways within complex networks. This path shows the specific sequence of actions a hacker might use to reach their goal (steal information) on a target computer. Meta-paths help cybersecurity experts understand these potential attack sequences and predict how attackers might try to breach a system.

As the relationships between nodes in heterogeneous graphs can have different meanings depending on meta-paths, it is difficult to directly apply traditional GNN methods. Traditional methods struggle to capture these nuanced relationships.

Definition 2: (Meta-path-based neighborhood): Given a node i and a meta-path Φ in a heterogeneous graph, the set of all nodes (including itself) connected to node i via the meta-path Φ is called the meta-path-based neighborhood of node i, denoted by $\mathcal{N}_i^{\Phi}$. A specific type of cyber attack may have a certain sequence of network traffic patterns. Meta-path-based neighborhood, in simpler terms, is a technique that can be used to identify these patterns and detect cyber attacks.

Definition 3: (Heterogeneous Graph of Cyber Threats): Cyber threats with different types of objects and relationships can be represented as a heterogeneous graph $G = (\mathcal{V}, \mathcal{E})$ with node type mapping $\Psi : \mathcal{V} \mapsto \mathcal{T}$ and edge type mapping $\psi : \mathcal{E} \mapsto \mathcal{R}$. Each node $v \in \mathcal{V}$ belongs to a specific object type in the set of node types $\mathcal{T}$, and each edge $e \in \mathcal{E}$ corresponds to a specific connection type in the set of edge types $\mathcal{R}$, where $|\mathcal{T}| + |\mathcal{R}| > 2$. The attributes of the nodes can be represented as an initial feature matrix X.

Definition 4: (Network embedding): Network embedding is a technique that turns intricate networks into simplified, numerical representations. This process captures the key connections and characteristics of the network, allowing these representations to be used for various tasks related to the network itself.

TABLE 1. The main notations of our model.

Nt.	Description
$\mathbf{h}_i$	The feature of node i
$\mathbf{M}_i$	Node type-specific transformation matrix
$\mathcal{G}_t$	Graph snapshot of $\mathcal{G}$ at the timestamp t
Φ	Meta-paths
P	The set of meta-paths $\{\Phi_1, \ldots, \Phi_P\}$
$\mathcal{N}i^{\Phi}$	The meta-path-based neighbors of node i
e_{ij}^{Φ}	Importance of node j for node i
$\mathbf{v}_i^{\Phi}$	The learned embedding of node i for the meta-path Φ
$V_t^{\Phi_P}$	Semantic-specific node embedding group
$\mathcal{E}_t^{\Phi_p}$	Commuting matrix of snapshot $\mathcal{G}_t$ under meta-path Φ_p
$\mathcal{V}$	The number of nodes
x_t	Unified global graph embedding of $G \in \mathcal{G}_t$

III. LITERATURE REVIEW

Strengthening the resilience of information systems is essentially based on detecting and preventing cyber threats. In recent years, innovative approaches and ideas have emerged, particularly in the use of machine learning and deep learning to detect and predict such threats.

In the quest to enhance information system security, predicting cyber attacks from heterogeneous and dynamic network data has emerged as a crucial endeavour. To address the inherent complexity of such networks, researchers have begun to explore advanced methodologies, including the use of meta-path-based graph neural networks.

These innovative approaches aim to capture the complex interaction between various cyber objects within systems. By leveraging meta-path-based graph neural networks, which excel at modelling heterogeneous and dynamic relationships, researchers seek to develop more robust and adaptive models for proactive cyber threat detection and mitigation. This nuanced understanding of network dynamics and heterogeneity promises to significantly advance our ability to prevent cyber threats and protect information systems against evolving risks.

TABLE 2. Dynamic heterogeneous and heterogeneous GNN models.

Heterogeneous			
Model	**Methodology**	**Aim**	**Ref.**
HGAT	Heterogeneous graph embedding techniques	Propose log2vec, a cyber intrusion detection model	[13]
Metapath2Vec	Meta-Path, Node Embedding, Random Walk	Used for node embeddings in heterogeneous networks.	[14]
Esim	Meta-Path, Random Walk	Used for similar node search in heterogeneous networks.	[15]
HAN	Meta-Path, Attention-based	Aggregates information from different-typed neighborhoods by utilizing both semantic-level and node-level attention mechanisms.	[16]
MBHAN	Random walk, Motif-level attention	Motif-based hierarchical heterogeneous graph attention network algorithm.	[17]
Dynamic Heterogeneous			
Model	**Methodology**	**Aim**	**Ref.**
DyHATR	Hierarchical Attention, GRU or LSTM.	Link prediction method	[18]
DyTriad	Graph Embedding	Leveraging the basic units to explore network dynamics and learn the graph embedding at different timestamps.	[19]
DHNE	Random Walk	captures dynamic semantic information.	[20]
CTP-DHGL	Meta-Path, Graph Embedding	Cyber Threat Prediction model based on Dynamic Heterogeneous Graph Learning	[2]
DyHAN	Graph Embedding, Hierarchical Attention	Dynamic heterogeneous graph embedding based on hierarchical attention	[21]
DyHNE	Meta-Path, Graph Embedding	Dynamic heterogeneous graph embedding model captures the dynamics between snapshots. Based on matrix perturbation theory	[22]
THGAT	Attention-based	Temporal Heterogeneous Graph Attention Network	[23]
HDGAN	Meta-Path, Attention-based	It is based on three levels of attention: structural, semantic and temporal attention.	[24]
HTGNN	Heterogeneous Temporal Graph Neural Network, Relation-r-based	It is a holistic framework tailored to heterogeneity with evolution in time and space for heterogeneous temporal graph representation learning.	[25]
ST-HGN	Spatial-Temporal Heterogeneous Graph Network, GRU for temporal modeling.	To model spatial and temporal dependencies in network data for proactive threat intelligence.	[26]
HRGCN	Unsupervised, Hierarchical Relation-augmented Heterogeneous GNN (HetGNN).	To detect anomalous graphs (e.g., system behaviors) in a set of heterogeneous graphs.	[27]
Our Work	node-level attention-based graph embedding, graph embedding based on meta-path level attention, and evolutionary pattern learning	It aims to effectively capture the complex relational structures and temporal dependencies in heterogeneous and temporally dynamic network data.	Soylu, Das, 2025

Suchacka et al. [12] proposed a neural network model to predict web-based bot networks by tagging active sessions and analysing historical usage data to identify malicious traffic patterns. Panker and Nissim [28] pioneered the use of volatile memory to detect malicious behaviour in unknown malware, extracting features and using machine learning models for automatic detection. Dionísio et al. [29] implemented a multitasking-based model for hunting cyber threat intelligence from tweets, while Lv et al. [30] designed hybrid kernel function (HKELM), a learnable model for detecting malicious cyber intrusions. Moodi et al. [31] introduced SSLPSO, a self-adaptive learning-based support vector machine model for detecting Android bot networks.

Similarly, Bao et al. [32] developed an automated system to uncover system risks by analysing event logs and predicting malicious execution behaviours.

Tang et al. [33] proposed a deep learning-based method to predict SQL injection attacks using ISP log data and statistical analysis. Leemans et al. [34] introduced an automatic tool for detecting hierarchical software threat events by characterizing threat patterns through software process sequences.

Inspired by the success of deep learning in pattern discovery from large-scale data, recent works have applied deep learning techniques to cyber threat prediction [35]. For example, Liu et al. [4] proposed log2vec, a cyber intrusion detection model based on heterogeneous graph embedding techniques. However, these approaches still rely heavily on domain knowledge and suffer from limitations in handling dynamic data.

In this context, the hierarchical attention mechanism within the DyHATR model has been examined by integrating it with both LSTM and GRU networks [18]. Comparative analysis revealed that GRU exhibited superior performance, particularly in terms of speed and efficiency. As a result, GRU was chosen as the optimal network for the hybrid model employed in this research.

Additionally, unlike the DyHATR model, our approach utilizes a more efficient node- and edge-level attention mechanism instead of the hierarchical attention mechanism, which allows for finer-grained and more targeted analysis of network interactions.

The Heterogeneous Graph Attention Network (HAN) model [16], which constitutes a part of our proposed framework, is not designed to operate on dynamic data. The superiority of the HAN model over these baseline models, which form part of other hybrid models such as DeepWalk, ESim, metapath2vec, HERec, GCN, and GAT, has been demonstrated [16]. In this work, we build on these findings to further explore the

advantages of HAN in the context of our proposed framework.

Our model offers significant advantages over the CTP-DHGL model by utilizing an attention-based node embedding technique [2]. While the CTP-DHGL model relies on attention-based node similarity measures for embedding, our approach directly employs attention mechanisms for node embeddings, resulting in more precise and nuanced representations of relationships between nodes. Furthermore, our model achieves a more lightweight and computationally efficient architecture by integrating GRU instead of Long Short-Term Memory (LSTM), as used in CTP-DHGL. This leads to faster training times and reduced memory consumption, particularly beneficial when working with large-scale datasets. These improvements enhance both the performance and scalability of our model.

To overcome these limitations, researchers have started to develop more advanced models such as meta-path-based graph neural networks [1], [2], [36] The objectives and methodology of some basic heterogeneous and dynamic heterogeneous GNN models are given in Table 2. These innovative approaches aim to better grasp the dynamics and complex structure of the network to be more robust and adaptive compared to traditional methods. By understanding the network's dynamics and complexity better, these new approaches have the potential to make us much more effective at stopping cyber threats and keeping information systems safe from new dangers.

Therefore, embracing novel methods in cybersecurity is crucial for fortifying information system security and staying ahead of ever-evolving threats. As such, utilizing cutting-edge techniques, like meta-path-based heterogeneous graph neural networks, is likely to be the driving force of future research in this domain.

IV. THE PROPOSED METHODOLOGY AND IMPLEMENTATION

Training of models created against cyber attacks is one of the most important steps directly affecting the performance and effectiveness of the model. In this study, we compare two different approaches for training our intrusion detection model: Firstly, the model is trained using datasets of past attacks that resulted in real damages. This allows the model to learn the typical behaviors and characteristics of real attacks.

After training is completed, the model performs analyses only through forward propagation on new incoming data. This method does not involve separate stages for training, testing, and validation. Backpropagation is costly in model training, and since only forward propagation is used in this method, it will run faster.

However, it will come with more significant disadvantages such as the inability to learn new information and adapt to changing threats.

In the second approach, the model will continually update itself by transferring what it learns from each instantaneous image to the next one using a neural network architecture like GRU. This way, the model will also learn new types of attacks and behaviors, resulting in a more dynamic structure. Through continuous learning, the model will become more resilient against constantly evolving threats and gain the ability to detect attack types it has not seen before. While this method may initially seem to require more resources due to the increased complexity of the model, considering the importance of data from systems affected by attacks and the evolving CPU and GPU technologies, this aspect will cease to be a disadvantage.

A. DATASETS

We evaluate our model using two datasets described below:

1) BETH

This dataset is among the largest publicly available resources in the field of cybersecurity, comprising approximately 8 million data points that monitor 23 distinct hosts. Each host engages in a variety of benign network activities, interspersed with at most one malicious attack instance. The scale and diversity of the data make it a valuable tool for evaluating the robustness and efficacy of intrusion detection systems (IDS). By containing both normal traffic and anomalous behavior, the dataset allows researchers to model and test a wide array of detection strategies.

The ability to analyze hosts engaged in typical, non-malicious activities—coupled with sporadic attack occurrences—provides a more realistic and challenging environment for security algorithms to operate within. This modern and comprehensive dataset contains modern host activities and attacks [37].

2) DARPA[1]

The 1998 DARPA Intrusion Detection Evaluation Dataset comprises 25,525 nodes and 4,554,344 edges, reflecting the complexity of the network topology and the multitude of interactions that occur within it. The attacks included in the

[1]DARPA: https://www.ll.mit.edu/r-d/datasets

Algorithm 1 The Algorithm of Our Proposed Model

```
Input: $G = \{\mathcal{G}_1, \mathcal{G}_2, \cdots, \mathcal{G}_t\}$ graph snapshots, $P = \{\Phi_1, \Phi_2, \cdots, \Phi_P\}$ meta-paths
Output: Predicted graph structure $\hat{\mathcal{G}}_t$
for $\forall \mathcal{G}_t \in G, t \in [1, 2, \cdots, T]$ do
    for $\forall$ meta-path $\Phi_m, m \in [1, 2, \cdots, P]$ do
        for $\forall i$ nodes do
            Calculate node-embedding with Eq: 4.
        end for
        With Equation 8, compute the importance of each meta-path $\Phi$.
    end for
    With equation 10, combine structural and semantic attention.
    Collect all instantaneous graph embeddings into a temporal array $X = \{x_1, x_2, \cdots, x_t\}$.
end for
for For each moment $t \in [1, 2, \cdots, T]$ do
    Calculate $z_t$ (Update Gate) using Equation 12;
    Calculate $r_t$ (Reset Gate) using equation 13;
    Using Equation 14, calculate $\tilde{h}_t$ (Candidate Hidden State);
    Calculate $h_t$ (Hidden State) using equation 15;
    Obtain the final predicted graph $\hat{\mathcal{G}}_t$.
    return $\hat{\mathcal{G}}_{t+1}$
end for
```

TABLE 3. Comparison of dataset characteristics.

Characteristic	BETH	DARPA'98
Node Count	Approx. 23 hosts	25,525
Edge Count	Approx. 8 million	4,554,344
Time Span	Modern	1998
Key Feature	Host-level activities	Network-level traffic
Attack Types	Interspersed attacks	4 main categories
Heterogeneity	High (various events)	High (IPs, ports, etc.)

dataset are diverse, ranging from denial of service (DoS) and probe attacks to more sophisticated user-to-root (U2R) and remote-to-local (R2L) exploits.

Each network event is labeled as either normal or malicious, making it ideal for supervised learning algorithms. Moreover, the sheer volume of data provides a robust platform for evaluating the scalability and performance of various detection methodologies, such as signature-based, anomaly-based, and machine learning-based intrusion detection systems.

The basic characteristics of the BETH and DARPA datasets used are presented comparatively in Table 3.

While the DARPA'98 dataset is considered chronologically old, it remains a fundamental benchmark in the intrusion detection literature. Using this dataset allows us to make direct and fair performance comparisons against numerous foundational and recent studies that also use this dataset. By evaluating our model on both classic, well-understood patterns from DARPA and more modern and complex activities from the BETH dataset, we demonstrate its robustness and adaptability across different eras of network threats.

B. DATA PREPROCESSING

At this stage, we apply preprocessing to the data obtained from log files. Using the *Python:pandas* library, we read the data, clean erroneous and missing entries in specified columns, and convert the columns into tensors. Next, we define relationships between the data and represent them as a homogeneous graph. Finally, we transform this homogeneous graph into a heterogeneous graph using the *Pytorch Geometric:HeteroData* [38] object, which allows for different types of nodes and edges to be represented in the graph.

From the 88 different attack types in the DARPA dataset, we selected the 10 most common for our analysis to ensure robust model training on well-represented classes. The scores in Table 4 reflect a baseline severity assessment based on common wisdom in the cybersecurity literature, where attacks with higher destructive potential are assigned higher scores.

However, a key feature of our approach is that these scores are designed as adjustable weights. This allows a security analyst to customize threat prioritization based on their organizational context or the evolving threat landscape. For example, an analyst could adjust the model's sensitivity to a specific attack type considered a high-priority threat to their system by increasing the score. The scores presented here represent our baseline configuration, but are intended to be flexible.

C. META-PATH SAMPLING

At this stage, we define and exemplify meta-paths on the heterogeneous graph obtained in the previous stage, utilizing the *pytorch-geometric:AddMetaPath* library. This method computes a feature vector for each sampled path by concatenating edge and node properties along the path.

TABLE 4. Scoring of attack types according to their harmfulness, with the lowest score being the most harmless.

Attack Type	Score	Description
neptune	0.9	denial of service attack
rootkit	0.9	unauthorized access
smurf	0.8	denial of service attack
pod	0.8	denial of service attack
worm	0.8	self-propagating malware
teardrop	0.7	denial of service on the target system by manipulating IP packets
warezmaster	0.7	gaining unauthorized access to a server to distribute malware
portsweep	0.7	a type of attack that scans the port looking for vulnerabilities
ipsweep	0.7	ICMP aims to collect IP addresses by sending ping queries
nmap	0.7	is a security tool used to scan target systems to identify open ports and services

We then use the *Python:sklearn:train-test-split* library to partition the calculated feature vectors into training, testing, and validation data, which serve as inputs to our model.

We use the meta-paths described below when training our model for two datasets.

- Attack $\longrightarrow$ Destination $\longrightarrow^t$ Attack
- Souce $\longrightarrow$ Destination $\longrightarrow^t$ Source

These meta-paths were chosen for their clear semantic significance in real-world threat detection scenarios:

- **Attack $\longrightarrow$ Destination $\longrightarrow$ Attack:** This meta-path captures patterns of repeated or coordinated attacks against a single target. It helps the model identify persistent threats or multi-stage attack campaigns targeting the same destination node.
- **Source $\longrightarrow$ Destination $\longrightarrow$ Source:** This path models reciprocal relationships and is crucial for identifying interactions like command-and-control (C2) communication or scanning activities where a source probes a destination and potentially receives a response. It highlights entities that are actively engaging with each other.

D. PROPOSED MODEL

The rise of artificial intelligence, with its constantly evolving cyberattacks, presents new challenges for traditional security methods built on identifying known threats. This makes us increasingly vulnerable to attacks unseen before. This highlights the critical need to predict unknown threats before they strike.

Considering the continuity of cyber attacks, heterogeneous networks can reach extremely large sizes. Therefore, we need to represent heterogeneous networks in a scalable way for real-time detection. Graph embedding is an effective and efficient way to represent large-scale networks [16], and thus we adopt graph embedding techniques to represent heterogeneous networks.

In this section, we propose a new semi-supervised hybrid GNN for heterogeneous networks. Our model follows a hierarchical attention structure: node-level attention $\rightarrow$ semantic-level attention $\rightarrow$ learning long-term temporal dependencies. Figure 3 illustrates the framework of our method and Table 1 shows the main notations of our model. Firstly, we propose node-level attention to learn the weights of meta-path-based neighbors and then perform semantic-specific node embedding by aggregating these weights. Subsequently, we evaluate long-term temporal dependencies using a gated recurrent unit.

1) NODE-LEVEL ATTENTION-BASED GRAPH EMBEDDING

It is evident that the neighbors of any node in a graph are important, and the meta-path-based neighbors of each node have different levels of importance [39]. Here, we use a node-level attention mechanism that can learn the importance of meta-path-based neighbors for each node in a heterogeneous graph and aggregate the importance of these meaningful neighbors to create a node embedding [16].

On a heterogeneous graph, different types of nodes have different feature spaces. To operate on our neural network that can learn from nodes connected by meta-paths, we need to map the features of different node types into the same feature space. For this mapping process, we need to create a specific transformation matrix $\mathbf{M}_i$ for each node type. The mapping process can be expressed as follows:

$$\mathbf{h}_i' = \mathbf{M}_i \cdot \mathbf{h}_i \tag{1}$$

Here, $\mathbf{h}_i$ represents the feature of node i, while $\mathbf{h}i'$ represents the reflected feature of node i. Thanks to the type-specific reflection process, the node-level attention neural network can handle arbitrary node types. We leverage self-attention to learn the weights between various node types [39]. Given a connected node pair (i, j) with meta-path Φ, the node-level attention e_{ij}^{Φ} can learn the importance e_{ij}^{Φ} of node j for node i [16]. The importance of the meta-path-based node pair (i, j) can be formulated as follows:

$$e_{ij}^{\Phi} = att_{node}\left(\mathbf{h}_i', \mathbf{h}_j'; \Phi\right) \tag{2}$$

Here, att_{node} denotes the deep neural network performing node-level attention. Given the meta-path Φ, att_{node} is shared across all meta-path-based node pairs. This is because there are some similar connection patterns under a meta-path. Equation (2) above shows that, given the meta-path Φ, the weight of the meta-path-based node pair (i, j) depends on their features. Here, the value of e_{ij}^{Φ} is asymmetric, meaning the importance of node i for node j and the importance of node j for node i are different. This is because the neighbors of two nodes may not be the same. This demonstrates that node-level attention can preserve the asymmetry, a critical feature of heterogeneous graphs.

Then, we only compute the e_{ij}^{Φ} value for the nodes $j \in \mathcal{N}i^{\Phi}$; where $\mathcal{N}i^{\Phi}$ represents the meta-path-based neighbors of node i (including itself). After obtaining the importance between

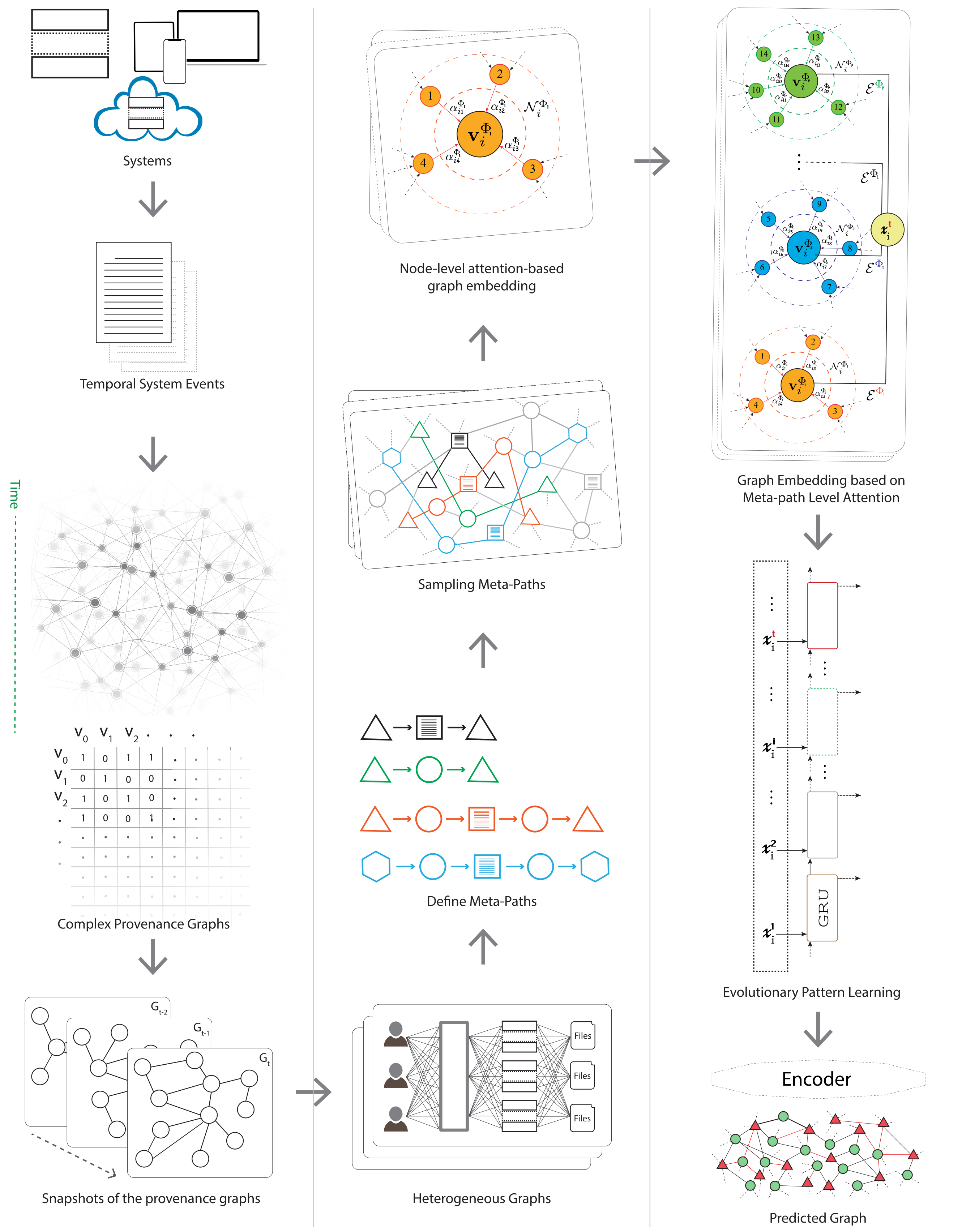

Systems
Temporal System Events
Time
Complex Provenance Graphs
Snapshots of the provenance graphs
Heterogeneous Graphs
Files
Define Meta-Paths
Sampling Meta-Paths
Node-level attention-based graph embedding
Graph Embedding based on Meta-path Level Attention
Evolutionary Pattern Learning
GRU
Encoder
Predicted Graph

FIGURE 1. General flow diagram of our model.

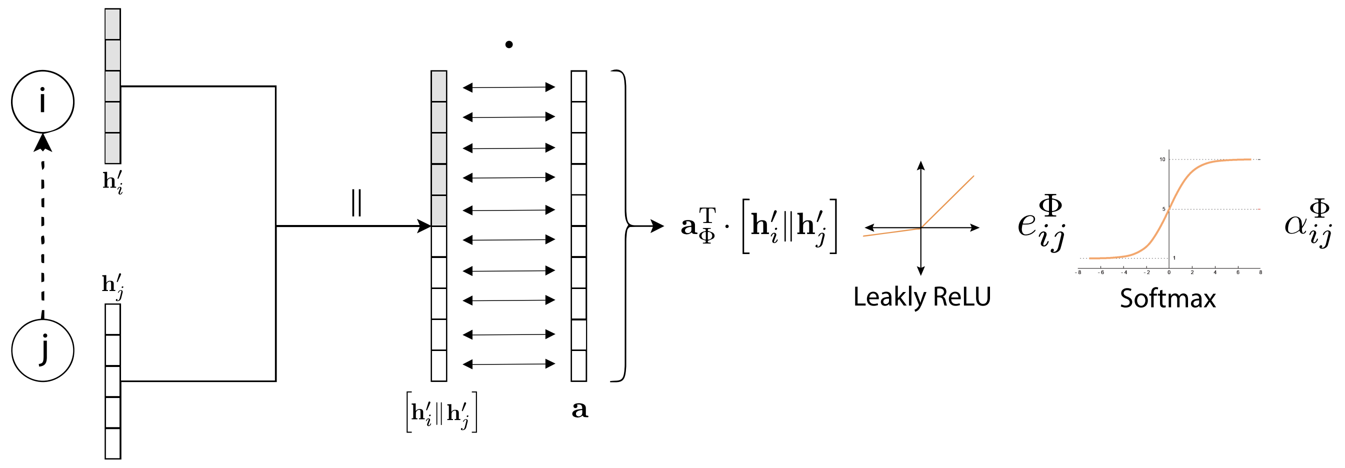

i
j
$\mathbf{h}'_i$
$\mathbf{h}'_j$
$\|$
$[\mathbf{h}'_i \| \mathbf{h}'_j]$
$\mathbf{a}$
$\mathbf{a}_\Phi^\mathrm{T} \cdot [\mathbf{h}'_i \| \mathbf{h}'_j]$
Leakly ReLU
e_{ij}^Φ
Softmax
α_{ij}^Φ

FIGURE 2. Gated recurrent unit cell.

meta-path-based node pairs, we normalize them through the softmax function to obtain the weight coefficient α_{ij}^Φ:

$$\alpha_{ij}^\Phi = \mathrm{softmax}_j \left(e_{ij}^\Phi\right) \tag{3}$$

$$\alpha_{ij}^\Phi = \frac{\exp\left(\sigma\left(\mathbf{a}_\Phi^\mathrm{T} \cdot \left[\mathbf{h}'_i \| \mathbf{h}'_j\right]\right)\right)}{\sum_{k \in \mathcal{N}_i^\Phi} \exp\left(\sigma\left(\mathbf{a}_\Phi^\mathrm{T} \cdot \left[\mathbf{h}'_i \| \mathbf{h}'_k\right]\right)\right)} \tag{4}$$

Here, σ represents the activation function, | denotes the concatenation operation, and $\mathbf{a}_\Phi$ represents the node-level attention vector for the meta-path Φ. The process steps are illustrated in Figure 2.

As seen in Equation 4, the weight coefficient of (i, j) depends on their feature. Here, α_{ij}^Φ weight coefficient is asymmetric, meaning they contribute differently to each other. This is because of the different normalization terms (denominator) due to the concatenation order in the denominator and having different neighbors. The α attention mechanism is a single-layer feedforward neural network parametrized by the weight vector $\mathbf{a}$ [40].

Then, the meta-path-based embedding of node i can be aggregated with the corresponding coefficients from the reflected features of its neighbors as follows:

$$\mathbf{v}_i^\Phi = \sigma\left(\sum_{j \in \mathcal{N}_i^\Phi} \alpha_{ij}^\Phi \cdot \mathbf{h}'_j\right) \tag{5}$$

Here, $\mathbf{v}_i^\Phi$ is the learned embedding of node i for the meta-path Φ. The embedding of each node is aggregated by its neighbors. Since the attention weight α_{ij}^Φ is generated for a single meta-path, it is semantically specific and can capture a type of semantic information. To make the training process more stable, we expand the node-level attention to multi-head attention [40]. Specifically, we iterate the node-level attention K times and combine the learned attentions as semantic-specific embeddings [16]:

$$\mathbf{v}_i^\Phi \longleftarrow \frac{1}{K} \sum_{k=1}^{K} \mathbf{v}_i^\Phi \tag{6}$$

So far, we have obtained the attention-based embeddings of node i for any heterogeneous graph, for the meta-path Φ. However, our cyber attack data obtained from a live system constantly changes over time and continuously increases. Therefore, there is a benefit in handling our data in real-time in terms of performance and success. Since our temporal heterogeneous graph has different snapshots for each time t, considering the meta-path set $\{\Phi_1, \ldots, \Phi_P\}$, for the graph at time t, we can obtain semantic-specific node embedding groups $\left(V_t^{\Phi_1}, V_t^{\Phi_2}, \ldots, V_t^{\Phi_P}\right)$.

2) META PATH LEVEL ATTENTION BASED GRAPH EMBEDDING

Given an instantaneous snapshot $\mathcal{G}_t \in G$ of a graph, we can effectively grasp the importance between nodes under a specific meta-path Φ_p, but fail to learn the importance of different meta-paths to represent global graph embedding. In fact, different meta-paths represent different semantic information and contribute differently to graph embedding. Here, we perform a graph embedding process based on meta-path level attention to learn the importance of different meta-paths and combine them to represent the global graph embedding of each graph snapshot $\mathcal{G}_t \in G$.

$$\left(\mathcal{E}_t^{\Phi_1}, \mathcal{E}_t^{\Phi_2}, \ldots, \mathcal{E}_t^{\Phi_P}\right) = \sigma_{\mathrm{path}} \left(V_t^{\Phi_1}, V_t^{\Phi_2}, \ldots, V_t^{\Phi_P}\right) \tag{7}$$

Here, $\mathcal{E}_t^{\Phi_p}$ represents the weight of meta-path Φ_p to represent the graph $\mathcal{G}t$. $\left(V_t^{\Phi_1}, V_t^{\Phi_2}, \ldots, V_t^{\Phi_P}\right)$ represents the set of node-level attention-based graph embeddings under different meta-paths, and σpath is a nonlinear activation layer based on the attention mechanism.

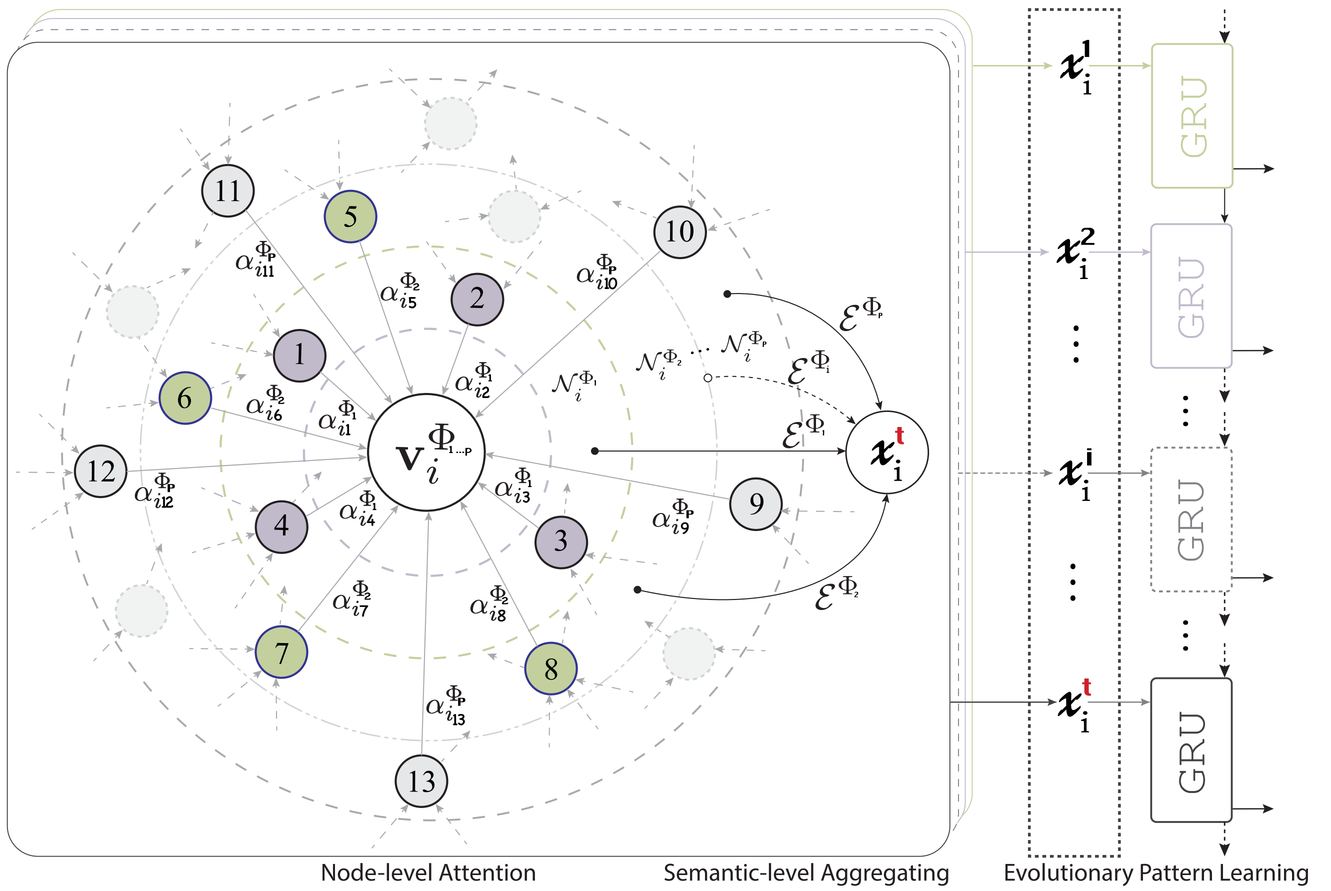


FIGURE 3. Our cyber threat prediction model.

To learn the importance of each meta-path Φ_p, we first calculate the importance of w_{Φ_p} as follows using a multi-layer perceptron (MLP) on graph embedding under a specific meta-path Φ_p.

$$w^{\Phi_p} = \frac{1}{|\mathcal{V}|}\sum_{i\in\mathcal{V}} q^T \cdot \tanh\left(\mathbf{W}\cdot\mathbf{v}_i^{\Phi_p} + b\right) \tag{8}$$

Here, $\mathbf{W}$ is a learnable weight matrix, $\mathcal{V}$ is the number of nodes, b is the bias vector, and q is the batch attention vector. After obtaining the importance of each meta-path, we normalize them using a softmax function.

$$\mathcal{E}^{\Phi_p} = \frac{\exp\left(w^{\Phi_p}\right)}{\sum_{p=1}^{P}\exp\left(w^{\Phi_p}\right)} \tag{9}$$

$\mathcal{E}^{\Phi_p}$ has the ability to interpret the contribution of the meta-path Φ_p to construct the global graph embedding representation $\mathcal{G}_t$, and the higher $\mathcal{E}^{\Phi_p}$, the more important Φ_p is. We can combine all meta-path-based embeddings by multiplying them with importance coefficients for the time t as follows:

$$x_t = \sum_{p=1}^{P}\mathcal{E}_t^{\Phi_p}\cdot V_t^{\Phi_p} \tag{10}$$

The result x_t in Equation (10) is an embedding matrix that combines structural and semantic attention. So far, we have obtained the aggregated global graph embeddings for each instantaneous $\mathcal{G}_t$ in G at different timestamps. The graph embeddings obtained from various graph snapshots can be summed up to form a time series $X = \{x_1, x_2, \ldots, x_t\}$.

Then, for semi-supervised node classification, we can minimize the Cross Entropy between the ground truth and predictions over all labeled nodes:

$$L = -\sum_{l\in\mathcal{Y}_L} \mathrm{Y}^l \ln\left(\mathbf{C}\cdot\mathbf{X}^l\right) \tag{11}$$

Here, C is the parameter of the classifier, $\mathcal{Y}_L$ is the set of indices of labeled nodes, Y^l and $\mathbf{X}^l$ are the labels and embeddings of labeled nodes, respectively. With the guidance of labeled data, we can optimize the proposed model through backpropagation and learn the embeddings of the nodes.

3) EVOLUTIONARY PATTERN LEARNING

Temporal evolutionary patterns depict the emergence and disappearance of nodes and edges over time. For the analysis of temporally changing data, recurrent neural networks offer promising performance, and hence, there has been increasing interest in these methods recently [2], [18]. We utilize a GRU model for the time series of graph embeddings obtained from graph snapshots $X = \{x_1, x_2, \ldots, x_t\}$. GRU is a simpler type

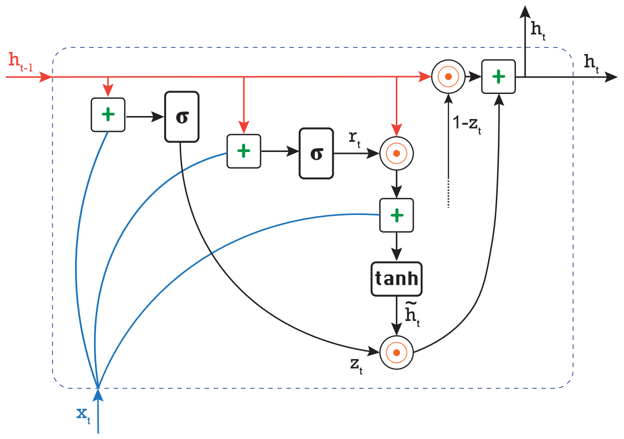


FIGURE 4. Gated recurrent unit cell.

of Long Short-Term Memory (LSTM) and has been introduced to capture long-term dependencies in many dynamic graph embedding models. GRU units have fewer parameters to learn compared to LSTM units, and as shown in Figure 4, each GRU unit contains only two gate cells, the update gate and the reset gate [41]. The temporal progression of GRU cells is illustrated in Figure 5. The state vector h_t for each input snapshot can be iteratively computed with the following equations (forward propagation equations for a GRU cell):

$$z_t = \sigma(W_z \cdot [x_t \| h_{t-1}] + b_z) \tag{12}$$

$$r_t = \sigma(W_r \cdot [x_t \| h_{t-1}] + b_r) \tag{13}$$

$$\tilde{h}_t = \tanh(W_h \cdot [x_t \| (r_t \odot h_{t-1})] + b_h) \tag{14}$$

$$h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \tilde{h}_t \tag{15}$$

Here; x_t is the input at the current time step. h_{t-1} is the hidden state at the previous time step. z_t is the update vector, determining how much of the old information to retain. r_t is the reset vector, determining how much of the old information to discard. $\tilde{h}_t$ is the candidate hidden state, representing the proposed updated hidden state. W_z, W_r, W_h are weight matrices, and b_z, b_r, b_h are bias terms, respectively. σ is the sigmoid function. $\odot$ denotes element-wise multiplication. | represents concatenation.

- Equation (12) defines the update gate. It controls how much of the current hidden state will be updated.
- Equation (13) defines the reset gate. It controls how much of the current hidden state will be forgotten.
- Equation (14) calculates the candidate hidden state. The new candidate hidden state is computed as a combination of the current hidden state and the output of the reset gate.
- Equation (15) defines the updated hidden state. The updated version of the current hidden state is calculated as a combination of the output of the update gate and the candidate hidden state.

E. ANALYSIS OF COMPUTATIONAL COST

To understand our model's computational demands, we can break down the cost of a single training epoch. Our analysis covers the entire sequence of T graph snapshots. We consider a graph with $|\mathcal{V}|$ nodes, P different meta-paths, and a total of $|\mathcal{E}_\Phi|$ meta-path instances. Other key variables are the transformed embedding dimension d', the K attention heads, and the GRU's hidden state dimension, h_{dim}.

The process involves three main stages:

- **Node-level Attention Embedding:** The first computational bottleneck is at the node level, where we calculate attention scores across all neighbors defined by the meta-paths. For any given time step, this operation runs across all P meta-paths and for each of the K attention heads. Modern libraries like PyTorch Geometric are highly optimized for this, using sparse matrix multiplications. As a result, the cost scales directly with the number of meta-path connections, leading to a complexity of $O(K \cdot |\mathcal{E}_\Phi| \cdot d')$ for one snapshot.
- **Meta-path Level Attention:** Next, the model must combine the embeddings generated from each of the P meta-paths. The most intensive part here is determining the weight of each meta-path, which requires looking at the embeddings of all $|\mathcal{V}|$ nodes. This brings the complexity of this step to roughly $O(P \cdot |\mathcal{V}| \cdot d')$ per snapshot.
- **Evolutionary Pattern Learning (GRU):** Finally, to understand how patterns change over time, we employ a GRU. For each time step, the GRU cell updates the embedding for every one of the $|\mathcal{V}|$ nodes. This cost is driven by the internal matrix multiplications of the GRU (involving its input, hidden state, and gates), resulting in a complexity of $O(|\mathcal{V}| \cdot (d' \cdot h_{dim} + h_{dim}^2))$.

Putting it all together, the total cost for one full epoch is the sum of these stages across all T snapshots. This gives our model an overall complexity of:

$$O\left(T \cdot \left((K \cdot |\mathcal{E}_\Phi| \cdot d') + (P \cdot |\mathcal{V}| \cdot d') + (|\mathcal{V}| \cdot h_{dim}^2)\right)\right)$$

The main takeaway is that our model's computational demand scales linearly with the number of snapshots, nodes, and connections. This linear relationship, powerfully handled by PyTorch's optimized operations, ensures our approach is practical and computationally feasible for large-scale, dynamic networks.

V. EXPERIMENTAL RESULTS

In this section, we introduce the metrics by which we measure the performance of our model. Then we share the results we obtained with the datasets.

A. EVALUATION METRICS

To quantitatively evaluate the threat type identification performance of our model, we use Macro-F_1 score and Micro-F_1 score as our evaluation metrics [42]. Table 5 gives the performance evaluation criteria. In Table 5, the 20%, 40%, 60%, and 80% columns indicate the proportions of the training data used to evaluate the model's performance across different metrics. These percentages represent the amount

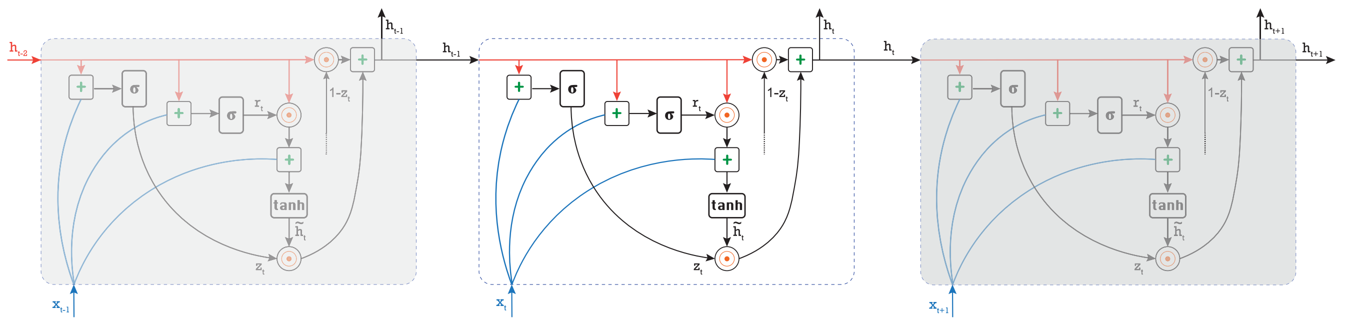


FIGURE 5. Gated recurrent unit.

of data utilized in each scenario. We apply 10-fold cross-validation and report the average performance measurements of Macro-F$_1$ and Micro-F$_1$ scores at a significance level of $\alpha = 0,05$. Macro-F$_1$ is a type of score that evaluates the average F_1 of all different labels in the hierarchy. TP_t, FP_t, FN_t represent true positives, false positives, and false negatives, respectively, for the t-th label in the label set L. Macro-F$_1$ is defined as follows:

$$\text{Macro-F}_1 = \frac{1}{L}\sum_{t\in L}\frac{2\cdot \text{Precision}_t \cdot \text{Recall}_t}{\text{Precision}_t + \text{Recall}_t} \tag{16}$$

$$\text{Precision}_t = \frac{TP_t}{TP_t + FP_t}, \text{Recall}_t = \frac{TP_t}{TP_t + FN_t} \tag{17}$$

Micro-F$_1$ is another type of F_1 score that takes into account the overall precision and recall of all labels. Micro-F$_1$ is defined as follows:

$$\text{Micro-F}_1 = \frac{2\cdot \text{Precision}\cdot \text{Recall}}{\text{Precision} + \text{Recall}} \tag{18}$$

$$\text{Precision} = \frac{\sum_{t\in L} TP_t}{\sum_{t\in L} TP_t + \sum_{t\in L} FP_t} \tag{19}$$

$$\text{Recall} = \frac{\sum_{t\in L} TP_t}{\sum_{t\in L} TP_t + \sum_{t\in L} FN_t} \tag{20}$$

B. TEST RESULTS

For our proposed hybrid model, we randomly initialize the parameters within a certain range, optimize our model with Adam [43], and set the learning rate to 0.005, the regularity parameter to 0.001, the size q of the semantic level attention vector to 128, the number of K per attention to 8, and the attention dropout to 0.6. We feed the data into the model cyclically according to the timestamp it contains. We also stop training if the validation loss does not decrease over 100 consecutive epochs. Table 5 presents the comparative performance results, where models are trained using different portions 40%, 60%, and 80% of the total available training data.

Graph neural network-based methods that combine structure and feature information, such as GCN and GAT, generally perform better [16]. The performance of our hybrid model on different datasets such as BETH and DARPA shows that its overall classification capabilities are strong. Considering the unstable nature of cybersecurity data, the F1-score combining Precision and Recall metrics best reflects the overall success of the model. The high Macro-F1 and Micro-F1 scores indicate that our model has a good balance of both precision and recall in the classification process. This indicates that your model works reliably in detecting and classifying cyber threats.

To provide a more granular insight into the model's classification performance beyond the aggregated F1 scores, we present the confusion matrices for both datasets.

Figure 6 shows the matrix for the multi-class classification task on the DARPA dataset. The diagonal elements highlight the number of correctly classified instances for each class,

TABLE 5. Performance comparison.

Datasets	Model	Metrics	Training Ratio		
			40%	60%	80%
BETH	SVM	Macro-F1	0.9651	0.9688	0.9695
		Micro-F1	0.9815	0.9842	0.9850
	Rand. Forest	Macro-F1	0.9693	0.9715	0.9722
		Micro-F1	0.9854	0.9879	0.9885
	HAN	Macro-F1	0.9718	0.9741	0.9750
		Micro-F1	0.9872	0.9885	0.9890
	DyHATR	Macro-F1	0.9785	0.9801	0.9808
		Micro-F1	0.9903	0.9918	0.9925
	CTP-DHGL	Macro-F1	0.9792	0.9809	0.9815
		Micro-F1	0.9910	0.9924	0.9930
	Our Work	**Macro-F1**	**0.9859**	**0.9855**	**0.9852**
		Micro-F1	**0.9954**	**0.9953**	**0.9951**
DARPA	SVM	Macro-F1	0.7812	0.7955	0.7980
		Micro-F1	0.8005	0.8143	0.8177
	Rand. Forest	Macro-F1	0.7998	0.8113	0.8150
		Micro-F1	0.8195	0.8298	0.8320
	HAN	Macro-F1	0.8177	0.8295	0.8341
		Micro-F1	0.8356	0.8471	0.8505
	DyHATR	Macro-F1	0.8601	0.8714	0.8755
		Micro-F1	0.8793	0.8889	0.8925
	CTP-DHGL	Macro-F1	0.8625	0.8749	0.8795
		Micro-F1	0.8788	0.8876	0.8912
	Our Work	**Macro-F1**	**0.9145**	**0.9135**	**0.9068**
		Micro-F1	**0.9226**	**0.9223**	**0.9167**

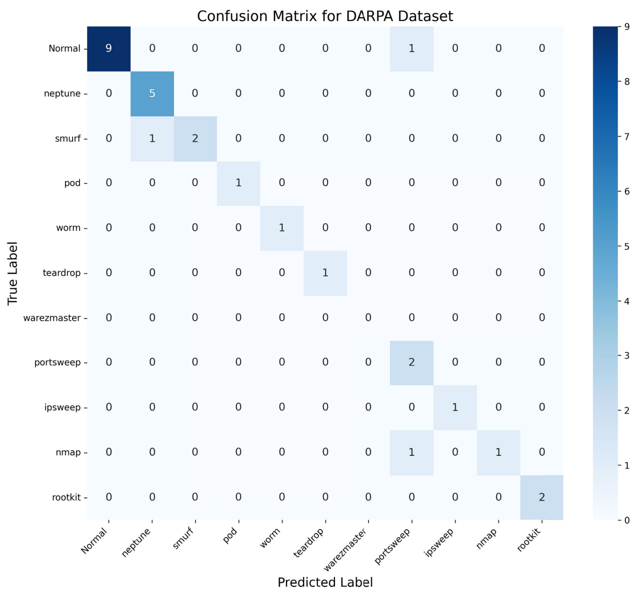


FIGURE 6. **Confusion matrix for our model on the DARPA dataset with an 80% training ratio.**

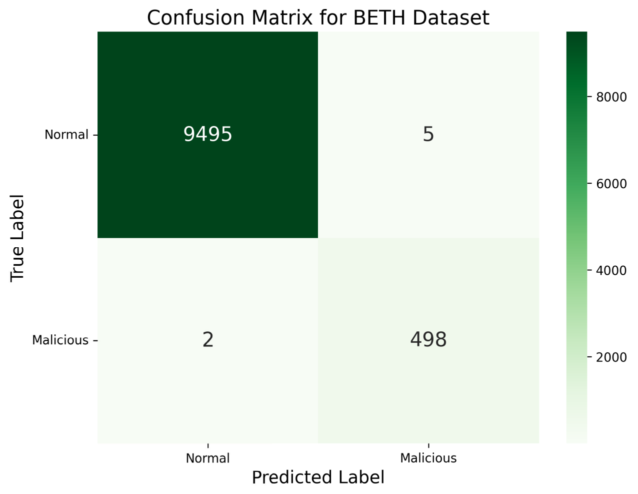


FIGURE 7. **Confusion matrix for our model on the BETH dataset with an 80% training ratio.**

while the off-diagonal cells reveal specific misclassifications made by the model.

As observed in the matrix, our model achieves a high rate of correct predictions, indicated by the strong diagonal concentration. The few misclassifications primarily occur between syntactically similar attacks (e.g., ‘neptune’ and ‘smurf’), which is a common challenge and demonstrates the model’s ability to discern even closely related threats with high accuracy.

Similarly, the confusion matrix for the BETH dataset, shown in Figure 7, demonstrates the model’s exceptional performance in a binary classification setting.

The matrix for the BETH dataset shows a very high number of true negatives (9495) and true positives (498), with a critically low number of false negatives (2). This indicates that the model rarely misses an actual threat, confirming the high precision and recall capabilities reflected in the F1 scores and validating its reliability for practical security applications.

VI. CONCLUSION AND FUTURE WORK

In this work, we developed **DyMHAG**, a hybrid GNN created specifically for predicting cyber attacks in dynamic and heterogeneous data. Its architecture uses a dual-level attention scheme on nodes and meta-paths to understand complex network structures, while a GRU tracks how these structures change over time. Testing on the BETH and DARPA benchmarks confirmed its effectiveness. Across our evaluations, **DyMHAG** performed better than established baselines like HAN, DyHATR, and CTP-DHGL. High F1-scores and our analysis of its classification behavior via confusion matrices support this conclusion.

Of course, the model is not without its limitations. Its predictive power is currently tied to the meta-paths we define manually, so it may not recognize entirely new attack strategies on its own. While it is computationally efficient for its complexity, applying it to web-scale networks in real-time would present a practical challenge requiring further optimization. Explainability is another open question. Like many neural networks, understanding the exact reasoning behind every single prediction from **DyMHAG** is not yet fully possible, which can be a barrier to trust for security teams.

The path forward for this research follows directly from these challenges. Our immediate goal is to work on an automatic meta-path discovery system to make **DyMHAG** more adaptive. We will also test its limits on newer, more complex datasets from modern IoT and 5G traffic, pushing its optimization for speed. Finally, a major focus will be on integrating XAI (explainable AI) methods. The aim here is practical: to make the model’s outputs not just accurate, but also transparent and actionable for security analysts who need to trust its recommendations.

ACKNOWLEDGMENT

This study is derived from Ph.D. dissertation entitled “Development of Novel Approaches Based on Geometric Deep Learning for Graph Visualization of Cyber Attacks” submitted at Fırat University, Graduate School of Natural and Applied Sciences, Department of Software Engineering under the supervision of Resul Daş.

ABBREVIATIONS

LSTM	Long Short-Term Memory
GCN	Graph Convolution Networks
GAT	Graph Attention Networks
GRN	Graph Recurrent Network
CNN	Convolutional Neural Networks
GRU	Gated Recurrent Unit
GNN	Graph Neural Networks

REFERENCES

[1] M. Lv, C. Dong, T. Chen, T. Zhu, Q. Song, and Y. Fan, "A heterogeneous graph learning model for cyber-attack detection," 2021, *arXiv:2112.08986*.

[2] J. Zhao, M. Shao, H. Wang, X. Yu, B. Li, and X. Liu, "Cyber threat prediction using dynamic heterogeneous graph learning," *Knowl.-Based Syst.*, vol. 240, Mar. 2022, Art. no. 108086. [Online]. Available: https://sciencedirect.com/science/article/pii/S0950705121011564

[3] J. Zhou, G. Cui, S. Hu, Z. Zhang, C. Yang, Z. Liu, L. Wang, C. Li, and M. Sun, "Graph neural networks: A review of methods and applications," *AI Open*, vol. 1, pp. 57–81, Jan. 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2666651021000012

[4] F. Liu, Y. Wen, D. Zhang, X. Jiang, X. Xing, and D. Meng, "Log2vec: A heterogeneous graph embedding based approach for detecting cyber threats within enterprise," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.* New York, NY, USA: Association for Computing Machinery, Nov. 2019, pp. 1777–1794, doi: 10.1145/3319535.3363224.

[5] A. Sperduti and A. Starita, "Supervised neural networks for the classification of structures," *IEEE Trans. Neural Netw.*, vol. 8, no. 3, pp. 714–735, May 1997. [Online]. Available: https://api.semanticscholar.org/CorpusID:5942593

[6] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini, "The graph neural network model," *IEEE Trans. Neural Netw.*, vol. 20, no. 1, pp. 61–80, Jan. 2008.

[7] A. Micheli, "Neural network for graphs: A contextual constructive approach," *IEEE Trans. Neural Netw.*, vol. 20, no. 3, pp. 498–511, Mar. 2009.

[8] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 26, C. J. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, Eds., Curran Associates, 2013. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2013/file/9aa42b31882ec039965f3c4923ce901b-Paper.pdf

[9] B. Perozzi, R. Al-Rfou, and S. Skiena, "DeepWalk: Online learning of social representations," in *Proc. 20th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2014, pp. 701–710, doi: 10.1145/2623330.2623732.

[10] A. Grover and J. Leskovec, "node2vec: Scalable feature learning for networks," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2016, pp. 855–864.

[11] W. L. Hamilton, R. Ying, and J. Leskovec, "Representation learning on graphs: Methods and applications," *CoRR*, vol. abs/1709.05584, 2017. [Online]. Available: http://arxiv.org/abs/1709.05584

[12] G. Suchacka, A. Cabri, S. Rovetta, and F. Masulli, "Efficient on-the-fly web bot detection," *Knowledge-Based Syst.*, vol. 223, Jul. 2021, Art. no. 107074. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0950705121003373

[13] T. Yang, L. Hu, C. Shi, H. Ji, X. Li, and L. Nie, "HGAT: Heterogeneous graph attention networks for semi-supervised short text classification," *ACM Trans. Inf. Syst.*, vol. 39, no. 3, pp. 1–29, Jul. 2021, doi: 10.1145/3450352.

[14] Y. Dong, N. V. Chawla, and A. Swami, "metapath2vec: Scalable representation learning for heterogeneous networks," in *Proc. 23rd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*. New York, NY, USA: Association for Computing Machinery, Aug. 2017, pp. 135–144, doi: 10.1145/3097983.3098036.

[15] J. Shang, M. Qu, J. Liu, L. M. Kaplan, J. Han, and J. Peng, "Meta-path guided embedding for similarity search in large-scale heterogeneous information networks," *CoRR*, vol. abs/1610.09769, 2016. [Online]. Available: http://arxiv.org/abs/1610.09769

[16] X. Wang, H. Ji, C. Shi, B. Wang, P. Cui, P. S. Yu, and Y. Ye, "Heterogeneous graph attention network," *CoRR*, vol. abs/1903.07293, 2019. [Online]. Available: http://arxiv.org/abs/1903.07293

[17] Q. Hu, W. Lin, M. Tang, and J. Jiang, "MBHAN: Motif-based heterogeneous graph attention network," *Appl. Sci.*, vol. 12, no. 12, p. 5931, Jun. 2022. [Online]. Available: https://www.mdpi.com/2076-3417/12/12/5931

[18] H. Xue, L. Yang, W. Jiang, Y. Wei, Y. Hu, and Y. Lin, "Modeling dynamic heterogeneous network for link prediction using hierarchical attention with temporal RNN," Tech. Rep., 2020.

[19] L. Zhou, Y. Yang, X. Ren, F. Wu, and Y. Zhuang, "Dynamic network embedding by modeling triadic closure process," in *Proc. AAAI Conf. Artif. Intell.*, vol. 32, no. 1, New Orleans, LA, USA, Apr. 2018, pp. 1–8, doi: 10.1609/aaai.v32i1.11257.

[20] Y. Yin, L.-X. Ji, J.-P. Zhang, and Y.-L. Pei, "DHNE: Network representation learning method for dynamic heterogeneous networks," *IEEE Access*, vol. 7, pp. 134782–134792, 2019.

[21] L. Yang, Z. Xiao, W. Jiang, W. Yi, Y. Hu, and H. Wang, "Dynamic heterogeneous graph embedding using hierarchical attentions," in *Proc. Adv. Inf. Retr., 42nd Eur. Conf. IR Res.*, Lisbon, Portugal, Apr. 2020, pp. 425–432, doi: 10.1007/978-3-030-45442-5_53.

[22] X. Wang, Y. Lu, C. Shi, R. Wang, P. Cui, and S. Mou, "Dynamic heterogeneous information network embedding with meta-path based proximity," *IEEE Trans. Knowl. Data Eng.*, vol. 34, no. 3, pp. 1117–1132, Mar. 2022.

[23] L. Zhang, J. Guo, Q. Bai, and C. Song, "Dynamic heterogeneous graph representation learning with neighborhood type modeling," *Neurocomputing*, vol. 533, pp. 46–60, May 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0925231223002096

[24] Q. Li, Y. Shang, X. Qiao, and W. Dai, "Heterogeneous dynamic graph attention network," in *Proc. IEEE Int. Conf. Knowl. Graph (ICKG)*, Aug. 2020, pp. 404–411.

[25] Y. Fan, M. Ju, C. Zhang, L. Zhao, and Y. Ye, "Heterogeneous temporal graph neural network," 2021, *arXiv:2110.13889*.

[26] J. Duan, Y. Luo, Z. Zhang, and J. Peng, "A heterogeneous graph-based approach for cyber threat attribution using threat intelligence," in *Proc. 16th Int. Conf. Mach. Learn. Comput.*, Feb. 2024, pp. 87–93.

[27] J. Li, G. Pang, L. Chen, and M.-R. Namazi-Rad, "HRGCN: Heterogeneous graph-level anomaly detection with hierarchical relation-augmented graph neural networks," 2023, *arXiv:2308.14340*.

[28] T. Panker and N. Nissim, "Leveraging malicious behavior traces from volatile memory using machine learning methods for trusted unknown malware detection in Linux cloud environments," *Knowl.-Based Syst.*, vol. 226, Aug. 2021, Art. no. 107095. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0950705121003580

[29] N. Dionisio, F. Alves, P. M. Ferreira, and A. Bessani, "Towards end-to-end cyberthreat detection from Twitter using multi-task learning," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2020, pp. 1–8.

[30] L. Lv, W. Wang, Z. Zhang, and X. Liu, "A novel intrusion detection system based on an optimal hybrid kernel extreme learning machine," *Knowl.-Based Syst.*, vol. 195, May 2020, Art. no. 105648. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0950705120301015

[31] M. Moodi, M. Ghazvini, and H. Moodi, "A hybrid intelligent approach to detect Android botnet using smart self-adaptive learning-based PSO-SVM," *Knowl.-Based Syst.*, vol. 222, Jun. 2021, Art. no. 106988. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0950705121002513

[32] L. Bao, Q. Li, P. Lu, J. Lu, T. Ruan, and K. Zhang, "Execution anomaly detection in large-scale systems through console log analysis," *J. Syst. Softw.*, vol. 143, pp. 172–186, Sep. 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0164121218301031

[33] P. Tang, W. Qiu, Z. Huang, H. Lian, and G. Liu, "Detection of SQL injection based on artificial neural network," *Knowl.-Based Syst.*, vol. 190, Feb. 2020, Art. no. 105528. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0950705120300332

[34] M. Leemans, W. M. P. van der Aalst, and M. G. J. van den Brand, "Recursion aware modeling and discovery for hierarchical software event log analysis," in *Proc. IEEE 25th Int. Conf. Softw. Anal., Evol. Reeng. (SANER)*, Mar. 2018, pp. 185–196.

[35] S. Wang and P. S. Yu, "Heterogeneous graph matching networks: Application to unknown malware detection," in *Proc. IEEE Int. Conf. Big Data (Big Data)*, Dec. 2019, pp. 5401–5408.

[36] S. Zhang, Z. Zhou, D. Li, Y. Zhong, Q. Liu, W. Yang, and S. Li, "Attributed heterogeneous graph neural network for malicious domain detection," in *Proc. IEEE 24th Int. Conf. Comput. Supported Cooperat. Work Design (CSCWD)*, May 2021, pp. 397–403.

[37] K. Highnam, K. Arulkumaran, Z. Hanif, and N. R. Jennings, "Beth dataset: Real cybersecurity data for anomaly detection research," in *Proc. ICML Workshop Uncertainty Robustness Deep Learn.*, 2021. [Online]. Available: https://api.semanticscholar.org/CorpusID:237258147

[38] M. Fey and J. E. Lenssen, "Fast graph representation learning with PyTorch geometric," in *Proc. ICLR Workshop Represent. Learn. Graphs Manifolds*, 2019, p. 1.

[39] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," *CoRR*, vol. abs/1706.03762, 2017. [Online]. Available: http://arxiv.org/abs/1706.03762

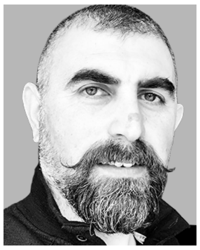
[40] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lió, and Y. Bengio, "Graph attention networks," vol. abs/1710.10903, 2017. [Online]. Available: https://api.semanticscholar.org/CorpusID:3292002

[41] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," 2014, *arXiv:1412.3555*.

[42] J. Opitz and S. Burst, "Macro F1 and macro F1," 2019, *arXiv:1911.03347*.

[43] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*.

MÜCAHIT SOYLU received the bachelor's and master's degrees from the Department of Mathematics, İnönü University, in 2007 and 2010, respectively. He is currently pursuing the Ph.D. degree in software engineering with Fırat University. Since 2010, he has been a Lecturer with İnönü University. Since 2008, he has been actively involved in the development of software projects for İnönü University. His current research interests include graph theory, software design and architecture, software quality and assurance, big data analysis, graph neural networks, and graph visualization.



RESUL DAS received the B.Sc. and M.Sc. degrees in computer science and the Ph.D. degree in electrical and electronics engineering from Fırat University, in 1999, 2002, and 2008, respectively.

From 2000 to 2011, he was a Lecturer with the Department of Informatics and concurrently worked as a Network and System Administrator with the university's IT Center. Since 2002, he has been an Instructor with the Cisco Networking Academy Program, delivering CCNA and CCNP courses. From September 2017 to June 2018, he researched as a Visiting Professor with the University of Alberta, Edmonton, Canada, under the TÜBİTAK-BİDEB 2219 Postdoctoral Research Fellowship Program. He also held the position of the Head of the Department of Software Engineering, from March 2020 to April 2023. He is currently a Full Professor with the Department of Software Engineering, Faculty of Technology, Fırat University. His research interests include computer networks, cybersecurity, the IoT and systems engineering, data science and visualization, and software quality assurance and testing. Globally recognized for his academic contributions, he was consistently listed among the top 2% of the "World's Most Influential Scientists," compiled by Stanford University researchers, for five consecutive years, from 2019 to 2024. He has served in editorial roles for several prestigious academic journals. He was an Associate Editor for IEEE ACCESS and *Turkish Journal of Electrical Engineering and Computer Science*. He is an Associate Editor for several Elsevier journals, including the *Internet of Things*, *Alexandria Engineering Journal*, *Telematics and Informatics Reports*, IEEE OPEN JOURNAL OF THE COMMUNICATIONS SOCIETY, and *International Journal of Grid and Utility Computing* (Inderscience).