UNIVERSIDADE FEDERAL DE PERNAMBUCO
CENTRO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIADA COMPUTAÇÃO

Luís Fred Gonçalves de Sousa

Detecting Unauthorized Access to Computer Networks Through Graph Transformers

Recife

2025

Luís Fred Gonçalves de Sousa

Detecting Unauthorized Access to Computer Networks Through Graph Transformers

This research is submitted to the Graduate Program in Computer Science at the Center for Informatics of the Universidade Federal de Pernambuco as a partial requirement for earning a Doctorate in Computer Science.

Research Area: Computational Intelligence

Advisor: Cleber Zanchettin

Recife

2025

.Catalogação de Publicação na Fonte. UFPE - Biblioteca Central

Sousa, Luis Fred Gonçalves de.
   Detecting unauthorized access to computer networks through graph transformers / Luis Fred Gonçalves de Sousa. - Recife, 2025.
   114 f.: il.

   Tese (Doutorado) - Universidade Federal de Pernambuco, Centro de Informática, Programa de Pós-Graduação em Ciências da Computação, 2025.
   Orientação: Cleber Zanchettin.
   Inclui referências.

   1. Cyber security; 2. Advanced persistent threats; 3. Graph neural networks. I. Zanchettin, Cleber. II. Título.

UFPE-Biblioteca Central

Luís Fred Gonçalves de Sousa

"Detecting Unauthorized Access to Computer Networks Through Graph Transformers"

Tese de Doutorado apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Pernambuco, como requisito parcial para a obtenção do título de Doutor em Ciência da Computação. Área de Concentração: Inteligência Computacional.

Aprovada em: 04/04/2025.

_____

Orientador: Prof. Dr. Cleber Zanchettin

BANCA EXAMINADORA

_____

Prof. Dr. Divanilson Rodrigo Campelo
Centro de Informática/UFPE

_____

Prof. Dr. Paulo Freitas de Araujo Filho
Centro de Informática/UFPE

_____

Prof. Dr. Flavio Arthur Oliveira Santos
Centro de Informática/UFPE

_____

Prof. Dr. Byron Leite Dantas Bezerra
Escola Politécnica de PE/UPE

_____

Prof. Dr. Eduardo Coelho Cerqueira
Faculdade de Engenharia da Computação e Telecomunicações do Instituto de Tecnologia/UFPE

Dedico este trabalho em memória de minha mãe, Maria Juscely.

AGRADECIMENTOS

Esta tese só foi possível porque pude contar com a assistência de pessoas que fizeram uma diferença bastante significativa na realização deste projeto de doutorado.

Gostaria de expressar minha imensa gratidão à minha querida esposa, e a nossos filhos, por todo o apoio que me permitiu concluir este trabalho. Agradeço, sobremaneira, ao meu pai, cuja inestimável assistência me permitiu chegar até aqui.

Um imenso obrigado ao meu orientador, Cleber Zanchettin, não apenas por ter compartilhado comigo seu valioso conhecimento, como também por ter acreditado em mim, mesmo muito antes de eu ter me tornado seu orientando. Sua contribuição foi extremamente decisiva para o desenvolvimento desta tese.

Agradeço também ao CNPq, pelo essencial apoio financeiro à minha pesquisa através do projeto MAI/DAI, e à empresa Tempest Security Intelligence, por ter apontado caminhos que se tornaram a base de todo este trabalho acadêmico.

Agradeço, ainda, a todos os meus companheiros de pesquisa, alunos e professores, do grupo MAI/DAI. Todas as conversas que tivemos e a intensa troca de conhecimento foram imensamente prazerosas e fundamentais para o desenvolvimento deste trabalho de pesquisa, bem como para o meu desenvolvimento acadêmico. Em particular, agradeço aos colegas Thiago Bispo, João Wojtyla, Paulo Matana, Júlio César, Henrique Arcoverde, bem como aos professores Divanilson Campelo e Paulo Freitas.

Por fim, mas não menos importante, agradeço aos demais amigos do Centro de Informática da UFPE, por seu duro trabalho e esforço para manter funcionando a infraestutura que deu suporte ao desenvolvimento do meu trabalho. Muito obrigado.

"Models are, by their very nature, simplifications. No model can include all of the real world's complexity or the nuance of human communication. Inevitably, some important information gets left out." (O'NEIL, 2016).

ABSTRACT

The proliferation of digital technologies, while enhancing productivity and access to new tools, has concurrently created opportunities for cybercriminals. This has led to a surge in digital abuses and cybercrimes, resulting in substantial losses for individuals, businesses, and governments. Advanced Persistent Threats (APTs) are central to many attacks, characterized by stealthy, gradual network infiltration to achieve objectives such as data theft and sabotage. Lateral movement, a decisive phase in APT campaigns, allows adversaries to consolidate their presence. Anomalous authentications serve as critical indicators of lateral movement, as they reveal intruder transitions between devices, often leveraging stolen credentials and exploiting vulnerabilities. Since computer network interactions form graph-structured data, graph-based algorithms, such as Graph Neural Networks (GNNs) and Graph Transformers (GTs), can be employed to detect anomalous interactions indicative of attacks within the computer networks. However, the effectiveness of these methods hinges on the representational power and performance of the graph models. Efficient node embedding aggregation in GNNs is pivotal for representing graph topology; existing simple aggregation methods (sum, mean, max) are limited, while the computational complexity of sophisticated approaches, such as Transformer-based methods, poses challenges for large graphs, despite their improved ability to capture long-range dependencies. Furthermore, many existing approaches neglect the temporal aspect of network events, which are inherently time-dependent. This work explores GNNs and GTs for unauthorized access detection in computer networks in two distinct experiments. First, we propose a link prediction approach incorporating a soft-attention mechanism to filter irrelevant node information during node representation aggregation. Second, we leverage recent advances in Transformer architectures for large graphs and propose a novel node classification approach for anomalous authentication detection that explicitly addresses the temporal dependencies between events at different granularities. The proposed models were trained on public datasets containing authentication logs from corporate networks. Experimental results showed that the proposed methods outperform state-of-the-art approaches in detecting anomalous authentications.

Keywords: Lateral movement detection, Advanced Persistent Threats, Graph Neural Networks, Cyber Security, Machine Learning.

RESUMO

A crescente adoção de novas tecnologias digitais tem ampliado as oportunidades para a prática de crimes cibernéticos, causando perdas significativas para indivíduos, organizações e governos. Entre as principais ameaças, estão as *Advanced Persistent Threats* (APT), nas quais os intrusos estabelecem um ponto de apoio inicial e expandem furtivamente sua presença na rede, acessando novos dispositivos e adquirindo mais informações sobre o alvo. O *Movimento Lateral* é uma etapa decisiva deste ataque, já que fortalece a presença do intruso na rede do alvo. Autenticações anômalas, frequentemente indicativas de alternâncias não autorizadas entre os dispositivos, são indicadores-chave desse estágio. A identificação de tais eventos tem o potencial de mitigar o movimento lateral e atrasar o avanço de um ataque em curso. Como as interações em redes de computadores formam grafos, algoritmos como Graph Neural Networks (GNN) e Transformers podem ajudar a identificar relações incomuns. No entanto, o sucesso na detecção de tais anomalias usando esses métodos está condicionado à sua capacidade de representação de grafos. A agregação eficaz de embeddings de nó em GNNs é determinante para representar devidamente a topologia do grafo, algo que muitos métodos ainda não alcançam plenamente. Além disso, a complexidade computacional de abordagens mais sofisticadas, como as baseadas em Transformers, é outro desafio em grafos de grande escala, apesar de melhorarem a captura de padrões em nós distantes. Neste trabalho, exploramos GNNs e Transformers no problema da detecção de acessos não autorizados em redes de computadores em dois experimentos complementares. Em um primeiro estudo, propomos uma abordagem baseada em predição de links entre os vértices, com um mecanismo *soft-attention* que facilita a agregação de representações de nó ao filtrar informações irrelevantes dos vértices. No segundo estudo, exploramos avanços recentes na literatura para a arquitetura transformer considerando grandes grafos e propomos uma nova abordagem baseada em classificação de vértices para detecção de autenticações anômalas. Essa abordagem permite considerar a dependência temporal entre os eventos em diferentes níveis de granularidade. Os modelos propostos foram avaliados em *datasets* públicos contendo registros de autenticação em redes corporativas. Os resultados experimentais mostraram que os métodos propostos superam as abordagens concorrentes do estado da arte na detecção de autenticações anômalas.

Palavras-chaves: Movimentação lateral, Aprendizagem de máquina, Redes Neurais de Grafos, Segurança digital, Ameaças digitais persistentes.

SUMÁRIO

1 INTRODUCTION

The increasing human dependence on new technologies and systems, such as 5G, cloud computing, and the Internet of Things (IoT), has brought cybersecurity to the forefront of public attention. Each new technology presents an opportunity for cybercriminals to exploit. With the evolution of attack techniques, well-organized, planned, and targeted cyber threats have become more frequent, often targeting critical infrastructures based on these technologies.

The COVID-19 pandemic brought us important moments of disruption, in which we witnessed several cyber attacks on users, private companies, and government (LALLIE et al., 2021) that should make us reflect on the need for our look at digital security. In particular, it is important to protect critical infrastructure against attacks by internal or external adversaries.

The evolution and massification of offensive techniques have allowed the execution of targeted attacks with losses to the companies involved and society in general. A representative example is the incident that deprived Brazilian citizens of access to vaccination data in 2021[1].

For the affected companies, these incidents resulted in significant financial losses and interruptions to their services and operations. Some recent incidents occurred within the space of a few weeks. For example, operators of a *ransomware*[2] have attacked the company JBS in June 2021. The Brazilian giant in the food sector suffered interruptions in its operations in the United States, Canada, and Australia. The incident compromised the food supply chain and sent meat prices soaring [3].

Still in mid-2021, the retailer Renner suffered a cyberattack by ransomware that compromised its operations in physical stores and e-commerce [4]. Equally impactful is the case of the Colonial Pipeline, the largest oil pipeline network in the US, which suffered a similar attack that resulted in significant losses. The company had its operations seriously compromised and still had to pay a millionaire amount to rescue the information that was in the possession of cybercriminals. The incident occurred in mid-2021 [5].

---

[1] Brazil health ministry website hit by hackers, vaccination data targeted. Reuters. December 10, 2021.

[2] a type of malware that employs advanced encryption features to restrict access to an infected system and charges a "ransom" before access can be re-established

[3] BBC. Meat giant JBS pays $11m in ransom to resolve cyber-attack. BBC News, 10 June. 2021.

[4] After a hacker attack, Renner denies that he paid $20 million to criminals. Available at: <https://exame.com/tecnologia/renner-sofre-ataque-de-ransomware-e-sistemas-da-empresa-ficam-fora-do-ar/>.

[5] Colonial Pipeline confirms data theft during ransomware. Available at: <https://www.tecmundo.com.br/seguranca/223178-colonial-pipeline-confirma-roubo-dados-durante-ransonware.htm>. Accessed on: 3 Jan. 2023.

In early 2022, the B2W e-commerce group suffered an attack resulting in a prolonged bottleneck of its services, seriously compromising the operation of important e-commerce platforms, such as Americanas S.A and Submarino. The unavailability of the platforms' services resulted in significant financial losses, also aggravated by the drop in the organization's share price on the stock exchange [6].

Even more important is the fact that adversaries have resorted to using **A**dvanced **P**ersistent **T**hreats (APT) to conduct cyber crimes, which is one of the main challenges of cybersecurity today. APT allows attackers to remain on the compromised network for extended periods and, usually unscathed, steal the data of organizations and governments (KHALEEFA; ABDULAH, 2022). Through sophisticated offensive techniques, adversaries remotely control machines compromised by them and extract confidential information of interest (ZIMBA et al., 2020). Generally, an APT has non-repetitive, unpredictable, and evasive behavior.

Due to this dynamic nature, cybersecurity software deployed across corporate networks based on traditional defense methods typically fails to detect this type of threat. Traditional security techniques indeed play a good role in analyzing rule-rich structured data. For example, *Security Information Event Management* (SIEM) software is widely used in detecting threats using rules. However, they still struggle to identify and discover sophisticated threats with unknown behaviors, such as advanced persistent threats. In those situations, solutions are needed that adapt to changes in the behavior of the attacker(s). Fundamentally, the adoption of adaptive solutions, such as machine learning techniques, in cybersecurity is justified by their ability to identify complex and subtle patterns and detect emerging threats that evade traditional rule-based or signature-based methods. ML models can learn from large volumes of data and continuously adapt to new malicious behaviors, improving existing defense systems (MINK et al., 2023).

Essentially, computer networks can be naturally modeled as graphs, in which nodes represent hosts and edges denote the communication links between them. This relational structure necessitates the use of machine learning methods that are capable of capturing and leveraging the inherent structural dependencies within the data.

Although machine learning methods have experienced an increasing adoption in cybersecurity domains (HALBOUNI et al., 2022; SARKER, 2023), most of the techniques are designed to

[6] Americanas and Submarino take websites offline again after a suspected hacker attack. Available at: <https://g1.globo.com/tecnologia/noticia/2022/02/20/americanas-e-submarino-tiram-sites-do-ar-apos-identificarem-acesso-nao-autorizado.ghtml> . Accessed on: 3 Jan. 2023.

operate on independent and identically distributed tabular data (i.i.d.), where each instance is represented as a fixed-size feature vector, and relationships between instances are absent or ignored. This fundamental assumption limits their ability to effectively capture and utilize the structural dependencies and interactions that are intrinsic to relational data, such as graphs. In graph-structured data, instances (nodes) are interconnected by edges, and the semantics of these connections often carry critical information for tasks such as node classification, link prediction, or anomaly detection.

When relational dependencies are ignored, as in the case of traditional machine learning methods, valuable information encoded in the graph topology, such as node neighborhoods or connectivity patterns, is lost. Such relational patterns may reveal coordinated attacks or anomalous communication behaviors that are imperceptible when instances are treated in isolation. Efforts to convert graph data into a tabular format typically result in the loss of expressiveness and context, rendering the learning process less effective. For example, two nodes with identical local attributes might play entirely different roles in a network depending on their connectivity, which cannot be captured through conventional feature-based representations alone. In contrast, Graph Machine Learning (Graph ML) techniques are explicitly designed to learn from relational structures, capturing patterns that are decisive for accurate predictions in graph-based contexts. This capability offers significant advantages when modeling complex attacks, such as Advanced Persistent Threats (APTs).

1.1 ADVANCED PERSISTENT THREATS

An APT is a sophisticated and premeditated cyber attack designed to persist and remain on the target system or network until its objectives are accomplished (SHARMA et al., 2023). Such objectives often involve stealing data, exfiltrating confidential information, or impeding an organization's critical operations through various attack vectors. As Tang et al. (2022) and Jaafer Al-Saraireh and Ala' Masarweh (2022) point out, the term APT was first introduced by the military and later adopted by the civilian IT security community. In 2006, the US Air Force had to deal with the challenge of discussing cyberattacks against its network with civilian experts. On the one hand, it was mandatory to avoid disclosing its findings about the origin of the perpetrators. At the same time, it was necessary to tell the specialists that these were attacks of a different type. So, they came up with the notion of Advanced Persistent Threats. Each word in the APT acronym carries an important meaning, from which we can derive a

description of the Vukalović e Delija (2015) attack:

- **Threat:** Because APTs intend to harm their targets, they pose a threat.
- **Persistent:** Attackers are persistent. They stay stealthy, slowly consolidating their presence by pivoting from one system to another within the organization's network, gaining useful information as they move. In parallel, they strategically export the acquired data to their command and control center. Typically, intruders stay for months in an attempt to compromise systems.

  Advanced: Attacks are usually coordinated by people with access to advanced resources and in-depth knowledge and are often well-funded. Incidents involving APTs, therefore, differ from traditional attacks.

APTs are highly complex and focused on specific objectives, utilizing intricate attack vectors. The development, deployment, and upkeep of APTs demand a substantial investment in workforce, IT infrastructure, and time. Consequently, an APT attack is expensive and usually requires backing from a well-resourced patron, typically a nation-state or a corporate organization. As pointed out by Steffens (2020) and Yang et al. (2021), most APTs are dedicated to spying on governments and companies. Far from being mere opportunistic cybercriminals, these are advanced, well-organized, and coordinated groups with sufficient resources to launch prolonged, sophisticated attacks (VUKALOVIć; DELIJA, 2015). In an APT attack, attackers typically employ social engineering tricks or vulnerability exploitation to gain initial entry into the network.

Since attackers aim to undermine critical services or steal data, they will need to move laterally[7] within the target's network to search for that data and gather information to help them progress the attack. So after initial entry, they maintain a presence in the target infrastructure and slowly gain a foothold, moving laterally and compromising one host after another within the organization's network.

To achieve its goals, an APT needs to take many intermediate steps. First, the threat must identify targets such as vulnerable devices or information about specific technologies. After these initial steps, attackers inject malicious software into a payload such as Microsoft Office

[7] Lateral movement is the tactic of moving within the network post-compromise. Refers to the methodical progression of an attacker through a compromised network in order to gain access to additional systems. This movement is often guided by reconnaissance activities into compromised computer networks, aimed at mapping the network topology and identifying vulnerable or poorly monitored machines.

documents, PDFs, or image files. After that, they usually use email attachments, URLs, or removable USB media to deliver infected files to the victim's host. Such a ruse allows the attacker to gain initial entry into the network.

Once clicked or opened, these infected files trigger the attacker's malicious code, which enters the system to exploit its vulnerabilities. Typically, attackers target vulnerable applications and user accounts or use an operating system feature that allows code to run automatically.

The complexity of this multi-layered process has led to the development of a concept known as the *Intrusion Kill Chain* (VELAZQUEZ, 2015), often referred to as the "Cyber Kill Chain" or simply the *kill chain*. The Intrusion Kill Chain was created to systematically categorize the various aspects of APTs, outline the stages that attackers typically go through, and enhance analysts' understanding.

The traditional Cyber Kill Chain does not explicitly cover lateral movement because its primary focus is on thwarting the initial phases of an attack. However, Lateral Movement (LM) is a critical phase during an APT attack and occurs after the initial phase when the attacker establishes an initial access point. During this stage, attackers seek to extend their presence across various systems and collect information about the target network's structure and architecture. Given its significance to the success of APTs, detecting lateral movement becomes decisive for countering an ongoing attack.

Authentication constitutes a fundamental component of cybersecurity, as highlighted by Pritee et al. (2024). Essentially, abnormal authentications are a critical indicator of lateral movement, as they can reveal transitions between devices that may lead to unauthorized authentication, often by stealing credentials. Bowman et al. (2020) used information derived from authentication event logs to build an unsupervised graph learning technique to detect lateral movement, and Paudel e Huang (2022) proposed a method to learn temporal node embeddings by using a graph embedding technique named CTDNE (NGUYEN et al., 2018) and then employs a gated-recurrent unit (GRU)-based auto-encoder top of node representations to learn long-term temporal information from authentication logs. The method can detect lateral movement upon identifying whether an incoming edge is anomalous or not.

However, despite their remarkable detection capabilities, many of these approaches only partially utilize the graph structure, extracting node features and then applying non-graph algorithms for further analysis. In contrast, Graph Neural Networks (GNNs) algorithms offer superior performance due to their complex structural capabilities, rendering graph embedding methods incomplete. In the cybersecurity domain, some authors have studied the use of GNNs

to perform lateral movement detection (KING; HUANG, 2023), fraud detection (DOU et al., 2020), and intrusion detection for IoT (LO et al., 2022). In particular, GNNs are multilayer neural networks that can learn patterns in structured graph data. Additionally, such algorithms eliminate the need for model retraining when new nodes are added, which is more appropriate for modeling dynamically evolving networks.

However, the success of detecting anomalous authentications using these methods depends on the representational power of these models. A critical challenge lies in how to aggregate node embeddings so that the GNN can better represent the network topology. In general, graph neural networks aim to learn permutation invariant hypotheses to have consistent predictions for the same graph when presented with permuted vertices/nodes, and such a property is achieved through neighborhood aggregation schemes. Existing approaches traditionally employ simple functions (e.g., sum, max, mean) on node embeddings to obtain consistent node representations. Nonetheless, we argue that an effective aggregation of node representations cannot be achieved through mere sum or mean operations. These aggregation functions can potentially lead to inaccurate and biased results by propagating irrelevant or ambiguous information. Therefore, filtering irrelevant node information during aggregation should improve model prediction.

Other significant challenges from existing GNNs pertain to the contextual range afforded by these algorithms and how this impacts the learning of patterns even from the most distant vertices of the graph. In particular, the inability to capture long-range dependencies hinders their performance. These limitations are especially pronounced in cybersecurity applications, where capturing global context and detecting subtle, long-range interactions is critical for identifying anomalous behavior, such as APTs (ALSHAMRANI et al., 2019).

Such limitations have been explored through generalizations of the Transformer architecture to graphs. The transformer architecture (VASWANI et al., 2017) has experienced an increasing research interest in the literature on graph neural networks in recent years, culminating in the Graph Transformer architectures (HENDERSON et al., 2023; SHIRZAD et al., 2023). It occurs because transformers are naturally graph neural networks (KIM et al., 2022; VELIčKOVIć, 2023). Concretely, the self-attention mechanism in the standard Transformer considers the input tokens as a fully connected graph, which is agnostic to the inherent graph structure of the data. Notably, one of the key advantages of Transformer variants for graph modeling is that they can learn long-range dependencies between nodes. Nevertheless, the high computational complexity of the Transformer architecture on large graphs is another challenge. Graph Transformers is a

relatively new and underexplored area of research.

Furthermore, many existing approaches overlook the temporal information associated with events in computer networks, even though these events are time-dependent. Analyzing graph structures without considering time-based perspectives may only reveal general trends that have limited relevance for cybersecurity threats. In reality, anomalies are rarely isolated events. To accurately identify them, we must consider the spatial relationships and the temporal context of an entity's interactions.

This thesis enhances GNN architectures inspired by Transformers (VASWANI et al., 2017) in two distinct experiments to detect unauthorized access in computer networks. Specifically, it aims to propose not only a novel model for detecting anomalous authentications—an important indicator of lateral movement—but also a method for identifying suspicious endpoints within the network, enabling their subsequent isolation. The Transformer architecture was chosen primarily for its ability to model long-range dependencies, in contrast to traditional GNNs, enabling it to capture critical information even from the most distant nodes in the graph. This characteristic is particularly advantageous in graph-based anomaly detection tasks, where relevant patterns may emerge from complex and non-local interactions across the network.

First, we propose a new link prediction approach incorporating a soft-attention mechanism to filter irrelevant node information during node representation aggregation. Second, we leverage recent advances in Transformer architectures for large graphs. We propose a novel node classification approach for anomalous authentication detection that explicitly addresses the temporal dependencies between events at different granularities. Experimental results on relevant datasets demonstrate that our proposed methods outperform competing approaches in anomalous authentication detection.

1.2 RESEARCH QUESTIONS

The central premise of our proposal is that users compromised in a lateral movement attack will interact with devices they usually do not have access to. Thus, the model profiles each network entity and distinguishes authentication activities that deviate from standard patterns. We argue that an effective aggregation of node features into a graph-level representation cannot be achieved through a simple sum or mean. Moreover, these aggregation functions can be sensitive to outliers, leading to inaccurate or biased results.

The research questions derived from this hypothesis are:

- What is the contribution of different aggregation methods to the quality of a GNN model when framing abnormal authentication detection as a link prediction task? How does the performance of that aggregation method compare to traditional ones (e.g., sum, mean, and max)?
- How does filtering out irrelevant node information from the graph affect the generalization of an anomalous login detection model?
- What is the role of each node in the network? Does incorporating temporal dynamics enhance the generalization capabilities of models designed for detecting anomalous nodes?

These questions explore the model's ability to correctly identify anomalous links between the entities and evaluate the efficacy of different aggregation schemes in improving the predictive performance of the model.

This study also explores the model's effectiveness in identifying anomalous nodes that signify suspicious hosts. It emphasizes the most relevant information about these nodes and their activities over time. Understanding the role of each node within the network is essential for detecting and isolating suspicious endpoints.

1.3 OBJECTIVES

This thesis aims to develop and validate Transformer-based models that operate on graph-structured data to detect lateral movement and suspicious endpoints within computer networks, an essential aspect of Advanced Persistent Threat (APT) attacks. Specifically, the study leverages Graph Transformer Networks to identify abnormal behaviors in network authentications. As discussed in the following chapters, we achieve these objectives by introducing key architectural innovations to enhance generalization ability. This capability could lead to the early detection of cyber threats and potentially minimize the damage caused by such attacks.

To accomplish this general goal, the following specific objectives are defined:

- Investigate and adapt Transformer-based architectures for processing graph-structured security data, emphasizing capturing long-range dependencies and heterogeneous node relationships.

- Propose architectural modifications to improve the generalization and robustness of Graph Transformer models when applied to the cybersecurity data domain.
- Develop a method to identify suspicious endpoints and lateral movement patterns using learned node representations and attention mechanisms derived from the proposed models.
- Conduct extensive experimental evaluations on real or realistic network authentication datasets, comparing the proposed approach against baseline methods in terms of accuracy and robustness.

1.4 ORGANIZATION OF THE DOCUMENT

The following chapters and sections detail how we plan to achieve our objectives. Chapter 2 examines the Cyber Kill Chain, a conceptual framework designed to delineate the sequential stages of a cyberattack, and explores the concept of lateral movement in cyber operations, providing an overview of current research developments in this area.

Chapter 3 presents the theoretical foundations of the thesis. It introduces key concepts in graph theory and graph machine learning and describes core prediction tasks: node classification and link prediction, framed within cybersecurity and lateral movement detection. The chapter proceeds with an overview of GNNs, the standard transformer architecture, and its adaptation to Graph Transformers (GTs). It concludes by discussing the main challenges in using GNNs for graph representation learning.

In Chapter 4, we propose a method for detecting abnormal logins, which can indicate lateral movement, by identifying anomalous links through graph transformers.

Chapter 5 presents a novel, scalable Graph Transformer (GT) model for detecting anomalous hosts in authentication logs through node classification. Addressing the scalability challenges of traditional GTs, the model incorporates temporal information and employs advanced attention mechanisms, including a soft-attention scheme to filter irrelevant data. Experimental results show superior performance compared to existing Graph Neural Networks (GNNs). Finally, in Chapter 6, we conclude this thesis, summarize our results, and present future research opportunities.

Chapter 6 provides the conclusion of this thesis, emphasizing the key findings, limitations, future research opportunities, and the published works stemming from this research.

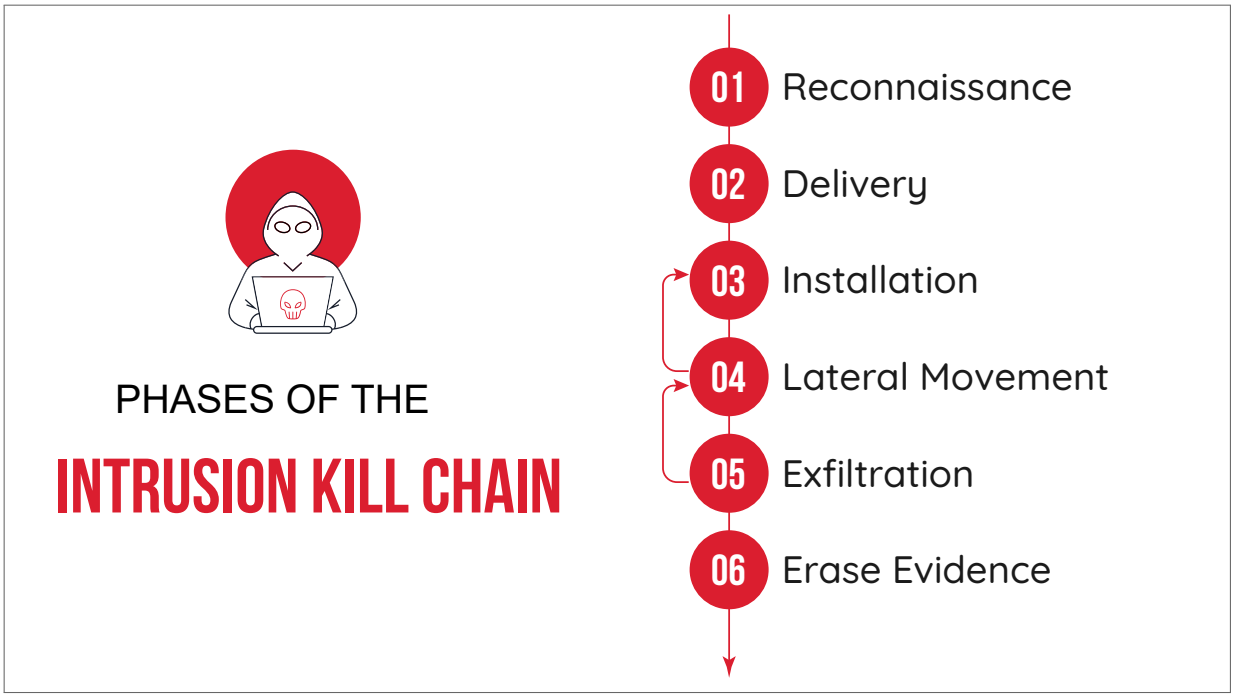
2 UNDERSTANDING ADVERSARY BEHAVIOR: INTRUSION KILLCHAIN AND LATERAL MOVEMENT

The Cyber Kill Chain is a conceptual framework created by Lockheed Martin to outline the sequential stages of a cyberattack. As discussed by (STEFFENS, 2020), this framework was adapted from traditional military kill chain concepts and has become a fundamental model in cybersecurity. Essentially, the Cyber Kill Chain helps to understand adversaries' behavior and effectively structure defensive measures (VELAZQUEZ, 2015).

Notably, the traditional Cyber Kill Chain framework does not explicitly cover lateral movement, as it primarily focuses on the initial phases of an attack. Specifically, this approach assumes that attacks will be detected and mitigated early, which is not always true. Attackers often successfully establish an initial foothold and move laterally across the network to compromise additional systems. Therefore, incorporating lateral movement into defense strategies is essential to address the full scope of an attack and mitigate its impact even after an initial breach has occurred.

2.1 INTRUSION KILL CHAIN

Figura 1 – The kill chain is a way to summarize and facilitate discussion about the steps taken by an adversary during an APT attack.



Source: Thesis author

Based on the Cyber Kill Chain, henceforth referred to simply as *kill chain*, Chen, Desmet e Huygens (2014) present a comprehensive study on APT attacks, characterizing the attack model and the analysis techniques typically adopted, in addition to presenting a clear distinction between APTs and traditional attacks. The authors detail how these persistent threats operate and formulate a typical attack process that comprises six phases, such as (I)*reconnaissance and weaponization*; (II) *payload delivery*; (III) *initial intrusion* (IV) *command and control* (V) *lateral movement*; and (VI) *data exfiltration*.

The phases described below are related to what was represented by Chen, Desmet e Huygens (2014), but with some abstractions to facilitate the presentation and the inclusion of the step *Erase evidence*, an important phase of the chain of events where the attacker erases his traces to make the work of forensic teams more difficult. Additionally, an illustration is provided in Figure 1:

- **Reconnaissance** - Attackers initially choose networks of organizations that may contain the information they are interested in. In this phase, relevant entities are searched, and information that can be used to facilitate later phases is collected.

- **Delivery** - Attackers inject malicious code into some payload such as Office documents, PDF files, images, or some utility software. Then, they deliver the infected file to the victim. A common method is to embed malicious code in a document sent to a recipient via email.

- **Installation or Infiltration** - Based on information collected in the reconnaissance phase and using social engineering tricks, the attacker infiltrates the organization's network to establish a foothold. In most cases, a well-crafted email with a malicious attachment or a link to a malicious website is sent to the target user. Once the user opens the attachment or clicks on the link, the malicious code is executed, and a backdoor program is installed on the user's system. At that point, a connection is established between the system and the attacker's remote server.

- **Lateral Movement** - Once the execution of the malicious code has been successful, the first computer on the organization's network has already been compromised, and the intruder can now control it. However, that initial device usually does not contain the information the attacker seeks. In fact, intruders often do not even know exactly where the relevant data is stored. However, having a computer under their control allows them

to move laterally across the network, pivoting to different systems until they find the information of interest. This phase can last several days or even months and is critically important to the adversary's goals of securing its presence on the target infrastructure. In *killchain*, lateral movement is described as an *loop*, as attackers continue to move laterally across the network throughout the entire operation.

- **Exfiltration** - The attackers have practically achieved their objective at this stage. When a significant fraction of the organization's systems are already compromised, they often only need to transfer data from the victim's network to their own systems. This is usually done through specific tools with *upload* functionality that the attacker installs on the victim's system. There are also cases where data is sent using legitimate tools already present in the compromised system. Moreover, the attacker could also install some *ransomware* to encrypt the stolen organization's data.

- **Erase Evidence** - The adversary wants to remain undetected during an APT campaign. Even the most sophisticated attacks can leave suspicious traces on compromised systems. Therefore, the most careful opponents will usually do their best to cover their tracks, whether during the final phase of the attack or even during all stages in between. This includes deleting any log data generated and removing any tools the attacker has installed as soon as it is no longer necessary to keep them on compromised systems.

Despite the Cyber Kill Chain providing an increased understanding of the anatomy of cyberattacks, the concept needed to be further developed to perform more solid threat modeling and threat assessments effectively. Based on this, the MITRE (ATT&CK, 2020) framework comprehensively describes cyber attackers' behavior once inside a computer network. The framework is continuously updated and is based on publicly known adversarial behavior.

Fang et al. (2022) suggests that understanding attack techniques and the purposes of each stage in an APT plays an important role in addressing these threats. Some of these techniques, it is worth mentioning, are highly decisive to the attacker's success. Network infiltration is only the first step for an attacker. Achieving his primary goals often requires exploring the network to find more important targets.

This suggests that lateral movement, which we will discuss in more detail in the next section, is the most critical stage in the lifecycle of an advanced persistent threat, as it allows adversaries to exploit and maintain their presence on the network and slowly acquire important

information about its structure and architecture. That said, it is reasonable to assume that early detection of lateral movement has the potential to retard an attack in progress Amin et al. (2021). In particular, by detecting the threat at this stage, it would be possible to reduce the severity of the damage caused significantly.

2.2 LATERAL MOVEMENT (LM)

As previously stated, the attackers' goal at this stage is to expand their presence to other systems and gather more information about the structure and architecture of the targeted network. Among the previously described kill chain phases, it is quite reasonable to state that lateral movement takes on a prominent position since it is a necessary step for the attacker to persist in the network for long periods. As pointed out by different experts, around 60% of attacks like those mentioned in the opening paragraphs of this work involve lateral movement (CANARY, 2020).

Typically, the machine used by the attacker to establish his initial presence on the network does not have the necessary user privileges to run some of the advanced tools he needs to consolidate his presence and install software, nor does it store the most valuable data he seeks. Therefore, one of the intruders' first steps is to identify other accessible machines and try to obtain the necessary login credentials to access those machines. The adversary will try to use various techniques to access other machines from an already compromised system. Most of the time, stolen legitimate credentials are used during this stage.

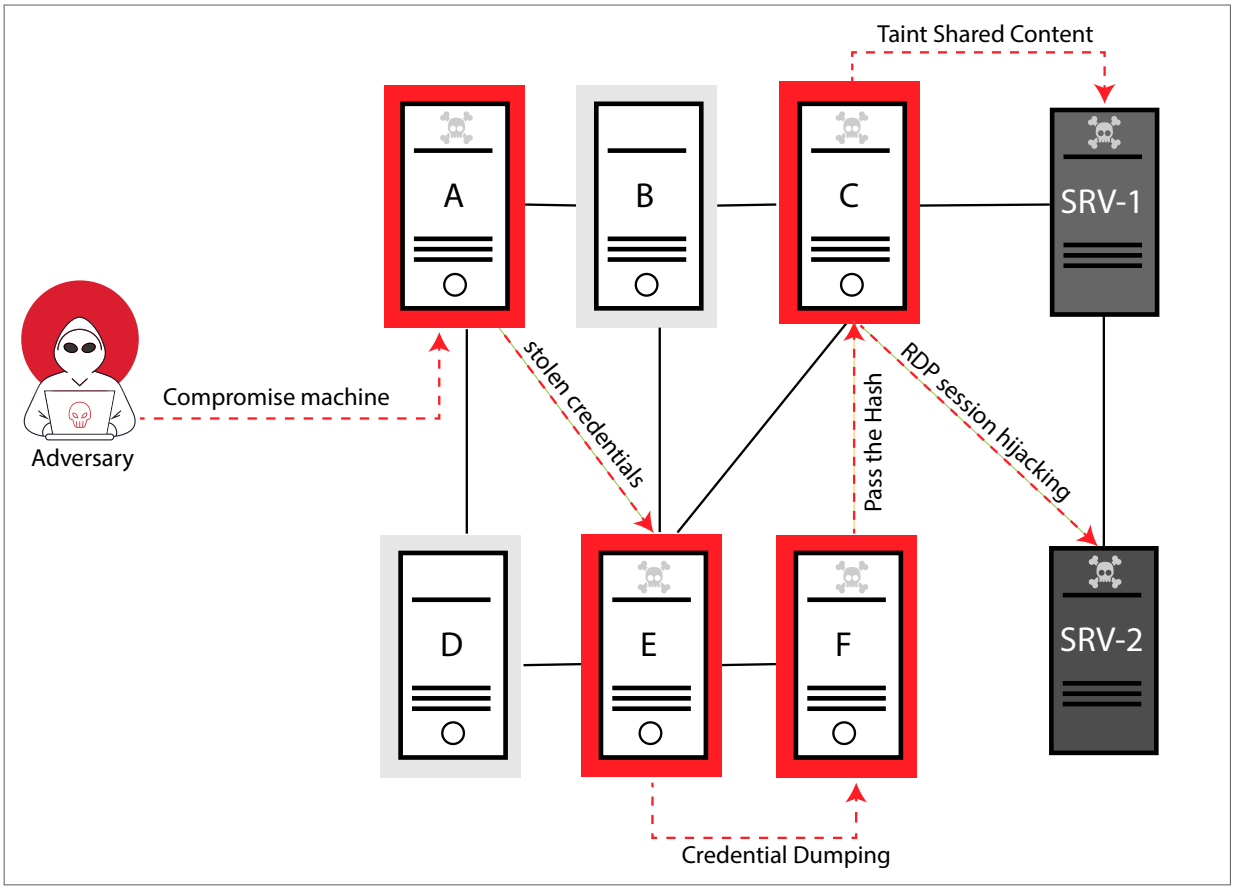
According to Alshamrani et al. (2019), LM can involve elevating privileges and, on other occasions, it consists of obtaining user passwords through *keyloggers*[1]. Eventually, the adversary may resort to techniques such as *pass-the-hash*[2] and/or exploitation of vulnerabilities in software. The chosen method usually depends on the target's system.

Adversaries often aspire to extended privileges within systems, typically at the administrator level. After elevating their privileges, the adversary searches the computer's memory for the credentials of other users, including administrators. With these credentials, attackers can log in to any computer on the network, even with administrative privileges. Afterward, the attacker can use legitimate administrative tools already installed by default on target machines.

[1] Keyloggers work in the background on the operating system and capture every user typing from the keyboard.

[2] occurs when an attacker steals a user's credential using some hash function.

Figura 2 – An illustration of lateral movement activity in a fictitious enterprise network with machines *A*,*B*,*C*,*D*,*E*,*F*, and servers *SRV-1* and *SRV-2*. The attacker first establishes an initial presence in *A* and then moves laterally to other machines, stealing credentials or hijacking remote sessions to reach higher-value machines within the network. By making logon in these machines, the adversary leaves traces. Authentication logs can reveal unusual logins typically associated with such lateral movement



Source: Thesis author

For example, PowerShell, which has been a default choice of attackers (LEMAY et al., 2018), is an efficient tool that allows the execution of codes at arbitrary times and the necessary commands for the attacker to switch between different machines along the network.

As they switch between different computers, attackers learn about the network structure and operation. Moreover, the threat actors typically seek users with access to interesting documents or weakly protected servers. At the same time, the attacker could install additional malware on strategically selected systems.

Eventually, the intruder could try to locate the Domain Controller (DC), a server responsible for storing the credentials of all users on the network. Moreover, the DC is typically used to verify that a user is authorized to log in to their desktop or when specific directories on a file server can be accessed. As pointed out by Steffens (2020), a compromised DC server irretrievably exposes all user credentials and allows the creation of additional accounts with

appropriate privileges.

Therefore, detecting lateral movement is decisive in mitigating the impacts of an APT, which, as aforesaid, include financial loss, theft of confidential data, and interruptions to essential services. However, in addition to these damages, which are already quite significant, the worst and most expensive consequence is the damage to the business's brand and reputation.

APT and, by extension, lateral movement pose some of the main challenges for traditional detection methods. To detect cyber threats and future abnormal behavior, threat detection systems mainly use heuristics and static signatures of known events, which are recorded in a large volume of data logs (HUBBALLI; SURYANARAYANAN, 2014; MASDARI; KHEZRI, 2020). Considering the high volume of data needed to create those signatures, the occurrence of an analyst error in the labeling process means rewriting a considerable number of rules.

Unfortunately, some of the techniques used by adversaries during lateral movement (e.g., stolen credentials) are too complex to formalize into signatures. During the attack, intruders will do their best to appear as legitimate users across the network. They will use the same tools as professional IT administrators, performing authentication on different machines within the network using legitimate user credentials, even if stolen from systems.

Furthermore, lateral movement does not follow an explicit or recurrent pattern, so the sequence of steps, tactics, and tools used varies according to the intruder's plans, who may adjust his methods periodically (POWELL, 2020). Therefore, the detection of malicious activities that characterize lateral movement must consider the behavioral characteristics of users along the network, and solutions must be required to adapt to changes in the attacker's behavior. Although commercial detection tools have evolved significantly over the last few years, they cannot incorporate the behavioral patterns of attackers into their operations yet.

Meanwhile, it has been highlighted in a recent study conducted by (MINK et al., 2023) that integrating defensive techniques with artificial intelligence is essential for advancing cybersecurity. With the rapid development of new machine learning-based technologies, defenders can now use artificial intelligence to improve the identification, response, and countermeasure of cyber attacks. From a broader perspective, researchers have recently delved into various machine learning approaches for anomaly detection.

Typically, anomaly detection algorithms are trained on normal system behavior during routine operation and then used to identify deviations of interest. In cybersecurity contexts, anomalies can be indicative of various malicious activities, such as fraud, cyber-attacks, or other forms of unauthorized access. Conventional algorithms like Isolation Forests, Autoencoders,

and others have successfully detected anomalous events (GONçALVES; ZANCHETTIN, 2024b; ALMEIDA et al., 2023). However, they often suffer from a limitation, which is treating individual events independently without considering their interrelated nature.

On the other hand, some of the existing approaches incorporate graph techniques that can be employed to address cybersecurity issues (BOWMAN; HUANG, 2021; PAUDEL; HUANG, 2022; POWELL, 2020; FANG et al., 2022). Regarding cybersecurity, data usually entails a group of interconnected entities. This can be seen in network activities, which typically take the form of a graph structure characterized by a set of nodes and edges. As such, the utilization of graph-based machine learning has the potential to bring about significant advancements in next-generation cybersecurity systems. Notably, several arguments and justifications have been presented by (BOWMAN; HUANG, 2021) to support the suitability of Graph Machine Learning for various domains within cyber-security, including LM detection. For example, Fang et al. (2022) designed an algorithm that can distinguish benign activity paths and lateral movement paths based on graphs and an unsupervised algorithm.

3 THEORETICAL FUNDAMENTATION AND LITERATURE REVIEW

This chapter provides the necessary theoretical background for the thesis. It begins by introducing fundamental concepts of graph theory and Graph Machine Learning, including graph definitions, properties, and early graph representation learning techniques like DeepWalk and Node2Vec. It outlines key prediction tasks, such as node classification and link prediction, contextualizing them within cybersecurity, particularly for lateral movement detection, and reviewing existing graph-based approaches. The chapter then delves into Graph Neural Networks (GNNs), explaining the message-passing framework and common architectures while also highlighting their limitations. Then, we discuss the standard Transformer architecture and its adaptation into GTs, mentioning their potential to capture global dependencies, overcome GNN limitations, and the associated challenges. We will discuss the link prediction task using GNN and conclude by highlighting the challenges of employing GNN for graph representation learning in the cybersecurity domain.

3.1 AN INTRODUCTION TO GRAPH MACHINE LEARNING

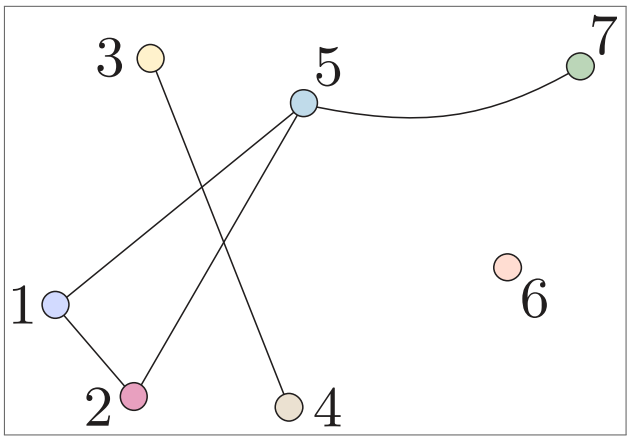
Graph Machine Learning, or shortly Graph-ML, involves employing machine learning methods to extract valuable insights from graph-structured data. These days, Graph-ML is a fast-growing research area. Due to the topic's relevance to the present thesis, this chapter briefly reviews the core concepts and literature on graph theory and graph representation learning through machine learning, which allows us to use the learned vectorial representations of graph structure to perform predictive tasks on network data, including anomaly detection.

3.1.1 GRAPH DEFINITIONS AND PROPERTIES

Graphs are a ubiquitous and versatile data structure that serves as a universal language for describing complex systems. In the most general view, graphs can capture structural information about a set of objects (or entities) and their relationships. Such objects are represented by *nodes* (or vertices) and their relationships by edges, as depicted in Figure 3. Graph theory is the branch of mathematics that studies interactions and connections. By examining the relationship between discrete objects, we can gain insights into the larger network in which

they exist. With our world becoming increasingly interconnected, it can be incredibly valuable to understand and predict the information a connection, or the lack thereof.

Figura 3 – A graph $\mathcal{G}$ whose node set is $\mathcal{V} = \{1, ..., 7\}$ and the edge set is $\mathcal{E} = \{\{1,2\},\{1,5\},\{2,5\},\{3,4\},\{5,7\}\}$.



Source: Thesis Author

For example, individual hosts (IP addresses) can be modeled as graph nodes to encode a computer network, and the communication between hosts is modeled as edges of the graph (LO et al., 2022). In cybersecurity, analyzing interactions using the graph theory lens is a robust tool against the adversary, as it offers a global view of the system (something that a cybercriminal usually does not have). In fraud detection systems, behavioral cues such as login times and locations can be altered by advanced fraudsters to camouflage their intents. On the other hand, it might be reasonable to argue that fraudsters could not have a global view of the entire network in which they are operating. It would, therefore, be impractical for them to adjust their behavior to fit as well as possible into this network without knowing all its structure and characteristics.

Definition 1. *Formally, a graph can be denoted as* $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ *where* $\mathcal{V} = \{v_1, v_2, \ldots, v_{|\mathcal{V}|}\}$ *represents the node set and* $\mathcal{E} = \{e_1, e_2, \ldots, e_{|\mathcal{E}|}\}$ *represents the edge set* $\mathcal{G}$*. The edge connecting two nodes* v_i *and* v_j *can be represented as* $e_{ij} = (v_i, v_j) \in \mathcal{E}$*. A node* v_i *is adjacent to another node if an edge exists between them.*

In many real-world applications, graphs constantly evolve as new nodes are added and new edges continuously emerge. These graphs are dynamic and can capture temporal information. In contrast, in static graphs, the connections between nodes are fixed.

Definition 2. *A dynamic graph* $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$ *is a graph whose node set* $\mathcal{V}$*, and the edge set* $\mathcal{E}$ *is dynamically changing with new nodes and edges added or removed. In a dynamic graph,*

timestamp information is associated with each node or edge to indicate when a connection emerges on the network.

We can describe a graph by listing the vertex and edge sets, which is accurate and technically correct. However, such a description alone is often insufficient and tends not to be particularly helpful in uncovering novel insights. A convenient way to obtain a codification of graph structure is to represent the graph through an adjacency matrix, which describes the connectivity between the nodes through a square matrix so that every node indexes a particular row and column and indicates when pairs of vertices are adjacent or not in the structure.

Definition 3. *Given a graph $\mathcal{G}$, the edge distribution can be denoted using an adjacency matrix $\mathbf{A} \in \{0,1\}^{N \times N}$. We can then represent the presence of edges as entries in this matrix. The (i,j)-th entry represents the connectivity $\mathbf{A}_{i,j}$ between nodes v_i and v_j. If $\mathbf{A}_{i,j} = 1$, there exists an edge, otherwise, $\mathbf{A}_{i,j} = 0$.*

Some graphs can also have weighted edges, where the entries in the adjacency matrix are arbitrary real values rather than $\{0,1\}$. For instance, a weighted edge in a machine-to-machine interaction over a computer network represented in a graph might indicate the frequency of the interaction between two machines.

Over the recent decades, the amount of graph-structured data available to researchers has substantially increased. However, the true potential of this data remains untapped, and the challenge is unlocking that potential. As these datasets grow in size and complexity, machine learning will be crucial in furthering our capability to model, analyze, and comprehend graph data.

However, the application of graphs in different computational solutions requires their structure to be encoded efficiently. In other words, we first need to obtain a useful graph representation. The progress made in obtaining graph representations is often closely tied to the concept of graph representation learning.

3.1.2 GRAPH REPRESENTATION LEARNING

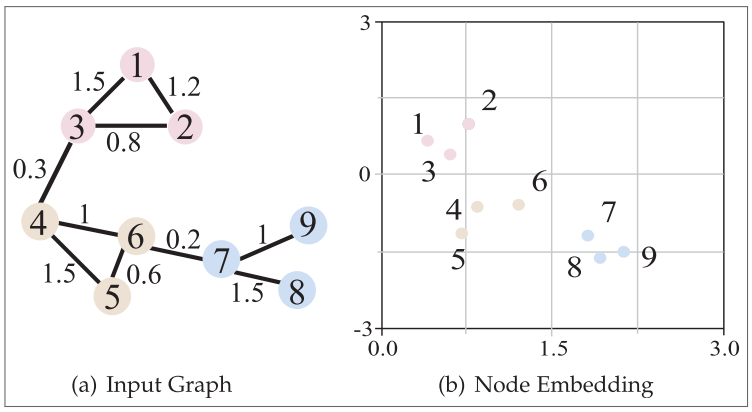
Graph Representation Learning refers to the development and refinement of algorithms and techniques that enable the automated learning of meaningful representations for graphs (HAMILTON; YING; LESKOVEC, 2017). By leveraging such approaches, relevant features and

patterns from graph-structured data can be extracted, and these representations can be used to perform prediction tasks, such as classification and clustering.

Definition 4. *Graph Representation Learning, also known as graph embedding, aims to learn a function* $\phi : \mathcal{V} \longrightarrow \mathbb{R}^d$ *that embeds the nodes* $v \in \mathcal{V}$ *in a graph into a low-dimensional Euclidean space where* $d \ll |\mathcal{V}|$.

As depicted in Figure 4, graph embedding allows us to automatically generate representation vectors for the graphs such that the output vectors preserve different properties of the graph, e.g., node proximities. Thus, the similarity of embeddings between nodes indicates their network similarity.

Figura 4 – A toy example of graph representation, with the input graph in (a) and the output node representations in (b).



3.1.2.1 Representing graph nodes as vectors

The concept of node embedding centers around creating vector representations of nodes by encoding them as low-dimensional vectors where geometric relations in this latent space correspond to relationships in the original graph. Node embedding involves two main steps. Firstly, an encoder model $\phi : \mathcal{V} \longrightarrow \mathbb{R}^d$ maps each node in the graph into a low-dimensional vector or embedding, learning a matrix $\mathbf{Z} \in \mathbb{R}^{|\mathcal{V}| \times d}$ containing the embedding vectors for all nodes, where $\mathbf{z}_v$ denotes the row of $\mathbf{Z}$ corresponding the embedding $\mathbf{z}_v \in \mathbb{R}^d$ of the node v. Secondly, a decoder model utilizes these low-dimensional node embeddings to reconstruct information about the nodes' neighborhoods in the original graph. For example, given a node embedding $\mathbf{z}_v$ of a node v, the decoder can predict the set of neighbors $\mathcal{N}(v)$ associated with that node.

The decoder plays a central role in reconstructing specific graph statistics from the node embeddings produced by the encoder. Although various decoders can be used, employing *pairwise* decoders is standard practice. For instance, the *Inner Product Decoder* and the *Bilinear Decoder* estimate the likelihood of an edge existing between two nodes based on the inner product of their embeddings, with the bilinear variant introducing a learnable weight matrix to model more complex interactions. The *Multi-Layer Perceptron (MLP) Decoder* leverages neural networks to capture non-linear relationships between node pairs. Meanwhile, *Distance-based Decoders* rely on similarity measures, such as Euclidean or cosine distance, to infer relationships based on geometric proximity in the embedding space.

These decoders can be viewed as estimating the similarity or relationship between pairs of nodes. A primary pairwise decoder may predict if two nodes are neighbors in the graph, equivalent to determining if an edge connects them. When a pair of embeddings $(\mathbf{z}_u,\ \mathbf{z}_v)$ are fed into the pairwise decoder, it produces a reconstruction of the relationship between nodes u and v. The objective is to optimize both the encoder and decoder to minimize the reconstruction loss, which can be expressed as:

$$DEC(ENC(u), ENC(v)) = DEC(\mathbf{z}_u, \mathbf{z}_v) \approx \textbf{Similarity}(u, v) \tag{3.1}$$

In sum, given a node u, we want to learn feature representations predictive of nodes in its neighborhood $\mathcal{N}(u)$. The **Similarity**(u, v) approximate the probability that nodes u and v co-occur on a random walk over the network.

Well-consolidated graph embedding methods, such as DeepWalk (PEROZZI; AL-RFOU; SKIENA, 2014), Node2vec (GROVER; LESKOVEC, 2016), and LINE (TANG et al., 2015), implicitly use decoders in the form of pairwise scoring functions as part of their training objectives. In particular, these methods employs inner-product when estimating node co-occurrences through decoders such that:

$$DEC(\mathbf{z}_u, \mathbf{z}_v) = \mathbf{z}_u^\top \mathbf{z}_v \tag{3.2}$$

Notably, the remarkable success of Word2vec(MIKOLOV et al., 2013a; MIKOLOV et al., 2013b) across several natural language processing tasks has aroused a growing interest in applying its Skip-gram model to learn node embeddings and adapting the inner-product approach to use stochastic measures of neighborhood overlap. DeepWalk (PEROZZI; AL-RFOU; SKIENA, 2014), which was the first method to make significant strides in this direction, considers the nodes in

a given graph as words in an artificial language, with sentences in this language generated by random walks – a random walk is an alternating sequence of vertices and edges whose edges are selected iteratively and random. DeepWalk subsequently uses the Skip-gram model from word2vec to learn node representations that maintain the node co-occurrence observed during these random walks.

Similar approaches such as Node2Vec (GROVER; LESKOVEC, 2016) and LINE (TANG et al., 2015) also achieved breakthroughs. For instance, the LINE model considered preserving both 1st-order and 2nd-order proximity between adjacent nodes. By doing so, it can successfully reconstruct proximity between nodes and achieve remarkable results in link prediction and node classification tasks. Node2Vec builds on the ideas presented in DeepWalk but offers greater flexibility and can capture a broader range of graph structures. Unlike DeepWalk, which uses a simple random walk strategy, Node2Vec employs a biased random walk strategy that balances between exploring and exploiting the graph structure. This strategy allows Node2Vec to capture both local and global graph structures. It enables the generation of embeddings that better represent the graph's topology[1].

However, as pointed out by Hamilton, Ying e Leskovec (2017), these methods have two significant limitations. Firstly, there is no parameter sharing between nodes in the encoder, leading to computational inefficiencies as the number of parameters increases linearly with the number of nodes. Secondly, direct embedding methods lack generalization capabilities, which means they cannot deal with dynamic graphs since they can only generate embeddings for nodes present during the training phase. These methods are problematic for many real-world applications where the underlying graphs evolve as new nodes or edges appear after the initial training phase.

3.1.3 PREDICTION TASKS ON GRAPHS

As pointed out by Hajiramezanali et al. (2019), once graph nodes have been mapped to a low-dimensional vector space according to the Definition 4, traditional feature vector-based machine learning formulations can be employed. Two typical node-focused tasks widely studied in this context are node classification and link prediction.

Suppose we have a vast e-commerce dataset containing millions of users (represented as nodes) and product reviews (represented as edges). Although a significant number of these

[1] Our use of "topology" to refer to local patterns is common in the network literature.

reviewers are genuine users of the e-commerce platform, there are also fraudulent reviewers among them. Dishonest companies often hire fake reviewers to post positive reviews of their products, which can harm the credibility of customer reviews and violate the e-commerce platform's terms of service. Detecting these fraudulent reviewers can be crucial for maintaining trust and transparency, but manually verifying each account would be a prohibitively expensive task. Therefore, we would like a model capable of classifying these accounts as fraudulent or genuine using only a limited number of manually labeled examples. This is a classic example of node classification where, in this example, the aim is to identify anomalous nodes in a graph (DOU et al., 2020).

Definition 5. Node classification – *In a graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$, some of the nodes are labeled, and these labeled nodes form the set $\mathcal{V}_l \subset \mathcal{V}$. The remaining nodes, without any label information, are referred to as the unlabeled set $\mathcal{V}_u = \mathcal{V} - \mathcal{V}_l$. It is worth noting that $\mathcal{V}_u + \mathcal{V}_l = \mathcal{V}$ and $\mathcal{V}_u \cap \mathcal{V}_l = \emptyset$. The primary objective of the node classification task is to predict labels for the nodes in $\mathcal{V}_u$.*

Let us now consider a computer network in which entities—such as users and machines—are represented as nodes, and authentication events between them are modeled as edges in a graph. A natural question arises: Can we predict the probability that a particular user will log into a given machine? More broadly, is it possible to employ machine learning techniques to infer the existence of edges between nodes in such a graph? This problem is commonly referred to as link prediction or relation prediction in the literature (ZHANG; CHEN, 2018a; Lü; ZHOU, 2011; KUMAR et al., 2020) and uses the knowledge from the existing relationships between entities to infer new relationships. However, we will simply call it *link prediction* in this work. Along with node classification, it is one of the more popular machine-learning tasks with graph data. It is worth mentioning that link prediction tasks can benefit applications in cybersecurity, such as detecting lateral movement by identifying anomalous edges in a graph (BOWMAN et al., 2020).

Definition 6. Link Prediction – *In a given graph $\mathcal{G} = \mathcal{V}, \mathcal{E}$, the edge set $\mathcal{E}$ comprises all observed edges. Let $\mathcal{M}$ denote the complete set of possible edges between nodes. The set of potential edges with unobserved edges between nodes is represented by $\mathcal{E}'$, where $\mathcal{E}' = \mathcal{M} - \mathcal{E}$. The objective of the link prediction task is to identify the edges that are most likely to exist in the graph. After link prediction, each edge in $\mathcal{E}'$ is assigned a score, which indicates its likelihood of existing or emerging in the future.*

In case of a temporal graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{T})$, $\mathcal{T}$ represents the timestamps associated with each edge. Each edge $e \in \mathcal{E}$ is thus a tuple (u, v, t), where $u, v \in \mathcal{V}$ are the nodes connected by the edge and $t \in \mathcal{T}$ is the timestamp indicating when the edge appeared. The objective of the temporal link prediction task is to identify the edges that are most likely to exist in the graph at a specific future time $t_f > t_{max}$, where t_{max} is the maximum timestamp in $\mathcal{T}$.

As aforesaid, after the models learn vector embeddings, various secondary tasks can benefit from such embeddings, including the widely studied node classification and link prediction. However, simply generating node embeddings and then employing machine learning algorithms to perform prediction is merely an incomplete use of graph structure. Furthermore, as large-scale graphs become increasingly complex, graph embedding models are being challenged to capture graph structures efficiently (HOANG et al., 2023). In response, there has been a growing body of research on deep graph neural networks capable of working with complex, large, and dynamic graphs (WU et al., 2021). Moreover, deep neural network-based models offer better generalization. They can more effectively capture relationships between entities and the overall structure of the graph, which motivated the rise of GNN architectures.

As previously discussed, graph embedding methods like Node2Vec and DeepWalk often combine multiple algorithms to generate node embeddings. The accuracy of the individual algorithms directly impacts the quality of these embeddings. In contrast, graph neural networks (GNNs) offer an end-to-end approach to generate node embeddings in graph-related tasks.

3.1.4 LATERAL MOVEMENT DETECTION THROUGH GRAPHS AND MACHINE LEARNING

APTs are a category of targeted attacks perpetrated by highly skilled technical adversaries that employ a broad spectrum of attack vectors for infiltration (ZIMBA et al., 2020). One of the critical characteristics of an APT is adapting to the defender's efforts to resist it Alshamrani et al. (2019). Moreover, APTs are stealthy and exhibit characteristics that are difficult to detect using most traditional rule-based defense tools.

According to Powell (2020), current commercial solutions that provide network intrusion detection and prevention on the network do not adequately protect against lateral movement. Notably, to defend against this threat, cyber defense tools need to adapt to the behavioral changes of the attackers, thereby increasing the research interest in machine-learning-based defense approaches. Unfortunately, the literature on lateral movement detection through machine

learning is still in its early stages of development. Notably, ideas that involve the application of graph-structured data have been emphasized (BOWMAN; HUANG, 2021). These approaches commonly represent internal network logins as motion graphs between machines or users and employ machine learning techniques to identify abnormal behavior, thereby detecting anomalies in graph-structured data.

In the wake of recent advances in machine learning, new techniques based on artificial intelligence have emerged with the potential to complement existing intrusion detection systems. For example, authors in (BAI et al., 2019) propose using an anomaly detection approach that utilizes Windows RDP event logs to identify indicators of lateral movement (LM) and explore different supervised machine learning (ML) methods for classifying RDP sessions. The authors recall that Remote Desktop Protocol (RDP) can be used during lateral movement to access unauthorized hosts successfully. Essentially, using RDP leaves footprints on both host and network logs.

Graph-based machine learning has the potential to make a significant impact on next-generation cybersecurity systems. In particular, the effectiveness of graph-based methods for anomaly detection in cybersecurity has gained significant traction, driven by their ability to capture complex relationships and patterns within network data. This capability allows them to effectively identify anomalous behavior, often indicative of malicious activity within a computer network (LAKHA et al., 2022).

For exampleBian et al. (2021) explores patterns extracted from authentication logs on the host using a graph representation of authentication events. By leveraging the patterns extracted from the graph, they evaluated various well-known machine learning algorithms to detect hosts affected by lateral movement. Subsequently, to reduce computational overhead and overfitting, the authors studied the correlation among the patterns and applied feature selection, which improved the performance for all tested anomaly detection techniques, with the Random Forest method achieving the best performance.

The work in (CHEN et al., 2018) explores a network embedding-based approach to identify potential malicious lateral movement behaviors. Firstly, they construct a communication graph of the host with patterns extracted from traffic logs or other data sources. The crucial assumption is that the graph consolidates information about the internal network structure and other important details from the utilized data sources. Next, they employ a learning method that considers the feature aggregation from vertices or their neighborhood through edge combination. Afterward, the authors trained a node classification model using graph embedding

methods. Specifically, a semi-supervised classification algorithm is trained on these embedding vectors to identify malicious hosts that indicate potential lateral movement.

In turn, Bian et al. (2021) explore graph-based features extracted from host authentication logs and propose a novel approach for detecting hosts targeted by LM during the LM phase of an APT attack. The authors then evaluate numerous ML classifiers, including ensemble techniques, to detect susceptible hosts.

The author in (BOHARA et al., 2017) argues that, during lateral movement, attackers typically establish a command and control (C&C) channel to guide their expansion process. The authors build a graph from host communication data to collectively analyze indicators of C&C and LM compromises. Then, they extract features from the graph to evaluate both the C&C and lateral movement-specific disruptions. Finally, the authors use the features to develop unsupervised anomaly detection methods using an ensemble of principal component analysis (PCA) and k-means on the lateral movement-related features, such that the model can find hosts that behave much like the infected devices on the network.

In turn, Powell (2020) proposed a new unsupervised method that identifies potentially malicious user logins. The authors approach the task as a graph anomaly detection problem, where individual vertices across a sequence of graphs representing a user's login history are tested for abnormal behavior. Using local graph measures to characterize each vertex, the authors employ the reconstruction error of a compression transformation applied to all vertices to identify unusual vertices.

Another interesting solution was explored by Bowman et al. (2020). In particular, the authors present an unsupervised graph learning technique for detecting lateral movement of Advanced Persistent Threats in enterprise-level computer networks. The method is unsupervised and utilizes industry-standard logging practices. The approach consists of an authentication graph and an unsupervised graph-based machine-learning pipeline. Crucially, their pipeline includes the offline training of node embeddings and a logistic regression link predictor. The link prediction task is a long-studied graph learning problem that aims to predict missing or future links between any pair of nodes. The detection of low-probability links in (BOWMAN et al., 2020) occurs by performing an embedding lookup for node pairs and running the link prediction on corresponding embeddings. A threshold is used, where links below a particular probability threshold will be forwarded to security experts for investigation.

More recently, the work in (FANG et al., 2022) proposed LMTracker, a method that uses event logs and traffic to establish heterogeneous graphs and generate representation vectors

for lateral movement paths. Afterward, the method employs an unsupervised algorithm to implement anomaly-based path detection.

Meanwhile, the authors in (PAUDEL; HUANG, 2022) suggest that the information flows and the lateral movement induced by the threat actor in the system deviate sufficiently from the expected behavior performed during legitimate activities across the network. To detect APT based on such abnormal behavior, the authors consider the anomaly detection problem in a dynamic graph whose goal is to detect anomalous edges. The proposed method was coined PICACHU and employed an unsupervised node embedding technique that captures both topological and temporal information from the graph. Then, a gated-recurrent unit (GRU)-based auto-encoder is used to learn long-term temporal information.

However, the previously mentioned approaches to detect lateral movement typically extract node features from graphs through the graph embedding methods and then utilize non-graph algorithms for further analysis. As discussed earlier, this is merely a partial use of the graph structure. Moreover, in recent years, large-scale graphs have posed challenges for many graph embedding models in effectively capturing complex graph structures due to their shallow architectures, as noted by Hoang et al. (2023). In particular, these methods can only generate embeddings for existing nodes during training, making them unsuitable for real-world applications where graphs evolve. In contrast, Graph Neural Networks (GNN) algorithms offer superior performance due to their complex structural capabilities and better adaptation to evolving networks (i.e., dynamic graphs).

3.1.5 GRAPH NEURAL NETWORKS (GNN)

Notably, research interest in GNNs has experienced substantial growth over the last few years (ZHOU et al., 2020; WANG; YU, 2022), with promising methods developed for a broad spectrum of domains. In the cybersecurity domain, to name a few examples, there are contributions approaching intrusion detection for IoT Lo et al. (2022), detection of software vulnerabilities (ZHOU et al., 2019a; CAO et al., 2021), lateral movement detection (SUN; YANG, 2022; PAUDEL; HUANG, 2022; GONçALVES; ZANCHETTIN, 2024a; SUN; YANG, 2022) on computer networks, and APTs (LIU et al., 2020; LI et al., 2021) related problems.

By leveraging GNNs, the models can learn to represent data as nodes with encoded relationships that standard approaches often overlook. This allows the algorithm to detect anomalies in individual processes and how they interact and influence one another. Specifically, by le-

arning from graph-structured data and leveraging local neighborhood information through message-passing mechanisms and downstream tasks as node classification (LIU et al., 2023), edge classification (LO et al., 2022; CAVILLE et al., 2022), and link prediction (ZHANG; CHEN, 2018b), GNNs can help us to identify malicious activities. For example, the work in (GONçALVES; ZANCHETTIN, 2024a) proposes to detect abnormal logins by uncovering anomalous links between nodes. The E-GraphSAGE algorithm (LO et al., 2022), which utilizes edge features and topological information, has improved intrusion detection systems through edge classification using GNNs.

GNNs are designed to learn functions on graphs and iteratively update the node representations by combining the representations of their neighbors and their representations. In particular, GNNs can be categorized into two types: **1)** *Spatial-based GNNs*, which perform spatial information aggregation involving neighbors nodes by operating directly on the graph structure (HAMILTON; YING; LESKOVEC, 2017; Justin Gilmer et al., 2017; CASANOVA; LIO; BENGIO, 2018; LO et al., 2022), and **2)** *Spectral-based GNNs* that leverage the spectral view of graphs by transforming the graph structure into the spectral domain using graph Fourier transform. Their primary focus is designing graph spectral filtering operators that filter specific frequencies of the input signal (DEFFERRARD; BRESSON; VANDERGHEYNST, 2016; YANG et al., 2022; WANG; ZHANG, 2022).

The propagation operator is an important component of both approaches. In particular, it is used to propagate information between nodes so that the aggregated information can capture both feature and topological information. Spectral methods define the propagation operator in the spectral domain, where a graph signal is first transformed to the spectral domain by the graph Fourier transform, and then the operator is employed.

Since spectral approaches are computationally more intensive (DEFFERRARD; BRESSON; VANDERGHEYNST, 2016) than spatial approaches, spatial graph-based techniques are usually preferred. However, as stated by Zhou et al. (2020), the major challenge of spatial approaches is defining the propagation operation with differently sized neighborhoods. This is because the propagation operation needs to aggregate information from different neighbors, and the neighborhood size can vary depending on the node.

As noted by Wu et al. (2021), since Graph Convolutional Networks (GCNs) (KIPF; WELLING, 2017) bridged the gap between spectral-based approaches and spatial-based approaches, the spatial-based methods have experienced significant advancements in recent times due to their appealing efficiency, flexibility, and generality.

3.1.5.1 GNN and the Message-Passing Framework

Before proceeding, it is important to recall the formal definition of a GNN. The following definition captures the core functionality of most GNNs.

Definition 7. *Formally, let $\mathbf{A} \in \mathbb{R}^{N \times N}$ represents the adjacency matrix from a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where N is the total number of nodes. Let $\mathbf{X} \in \mathbb{R}^{N \times C}$ denote the node attribute matrix, where C is the number of features for each node. Moreover, let $\mathbf{H} \in \mathbb{R}^{N \times F}$ denote node representations, where F is the dimension of node representations. Graph neural networks aim to learn effective node representations $\mathbf{H}$ by combining the graph structure information and the node attributes, which are further used for secondary tasks such as link prediction and node classification.*

We have two important functions in each layer starting from the initial node representation $\mathbf{H}^0 = \mathbf{X}$. The first one is referred to as **AGGREGATE**, which aims to aggregate information from the neighboring nodes of each specific node. The latter is **COMBINE**, which updates the node representations by merging the aggregated information from the neighbors with the current node representations. Crucially, we can define the general framework of GNNs using Algorithm 1:

Algorithm 1 Essentially, GNNs iteratively update the node representations by combining the representations of their neighbors and their own representations.

$\mathbf{H}^0 = \mathbf{X}$ // Initialization
for $k = 1, ..., K$ **do**
  $a_v^k = \textbf{AGGREGATE}^k\{H_u^{k-1} : u \in \mathcal{N}(v)\}$
  $H_v^k = \textbf{COMBINE}^k\{H_v^{k-1}, a_v^k\}$
end for

where $\mathcal{N}(v)$ is the set of neighbors for the v-th node, and $\mathbf{H}_v^k$ is the final node representations in the last layer.

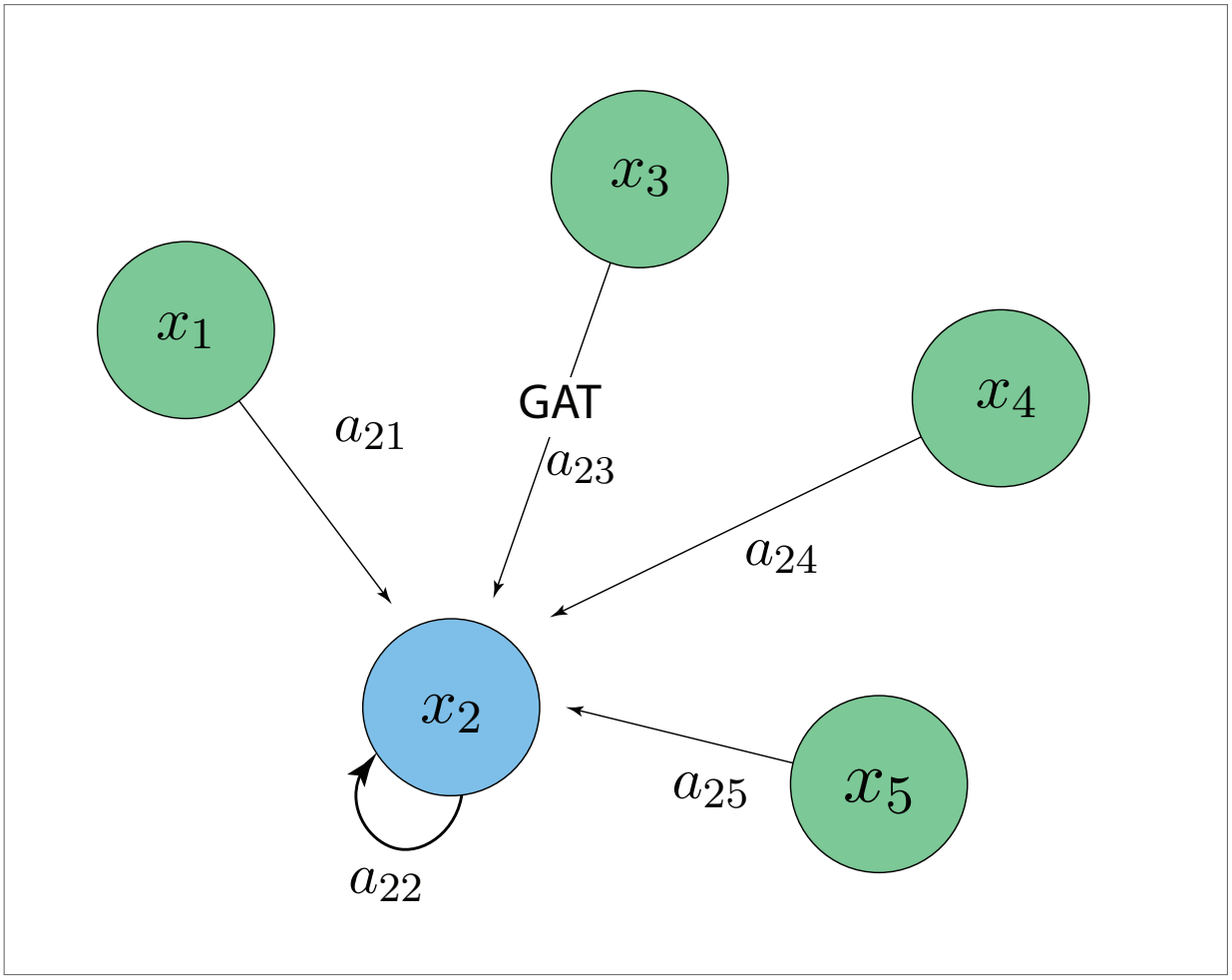
Important GNN models include Graph Convolutional Networks (GCN) (KIPF; WELLING, 2017), Graph Attention Neural Networks (GAT) (CASANOVA; LIO; BENGIO, 2018), Neural Message Passing Networks (MPNNs) (Justin Gilmer et al., 2017) and GraphSAGE (HAMILTON; YING; LESKOVEC, 2017).

The GCN is a popular graph neural network architecture (WU et al., 2021). This is attributed to their simplicity and effectiveness across various tasks and applications. However, as discussed in (CASANOVA; LIO; BENGIO, 2018), edge weights in GCNs may not be able to reflect the true

strength between two nodes. Such limitation motivated the rise of the GAT. Fundamentally, GAT tries to learn the importance of each node's neighbor based on the attention mechanisms.

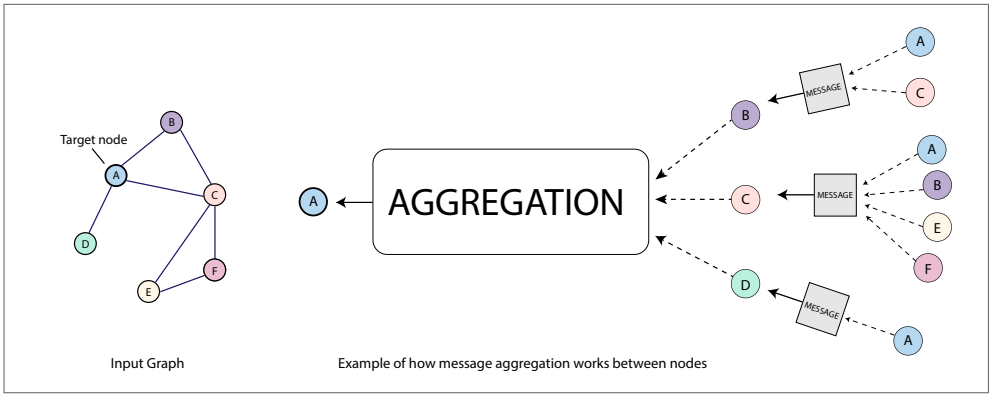
Specifically, compared with other operators, such as GCN and SAGE, GAT incorporates the attention mechanism into the propagation step. In GATs, the hidden states of each node are computed by attending to its neighbors, following a self-attention strategy (CASANOVA; LIO; BENGIO, 2018). This means that different weights are assigned to each neighbor, depending on its importance to the node's representation, as depicted in Figure 5. This allows GATs, theoretically, to alleviate noise and achieve better results than previous methods. GATs also utilize the multi-head attention mechanism, first introduced by Vaswani et al. (2017). This allows GATs to learn multiple independent attention head matrices concatenated or averaged to compute the hidden states. This further stabilizes the learning process and improves the performance of GATs.

Figura 5 – GAT tries to learn the importance of each neighbor through a *node-level attention mechanism* a that measures the attention coefficients for any pair of nodes.



Source: Thesis author. Adapted from Casanova, Lio e Bengio (2018)

Figura 6 – A MPNN aggregates node messages from its local neighborhood. This process recurs as messages are collected from neighbors' neighborhoods, forming a tree-like computation graph. The tree structure illustrates the hierarchical message aggregation in the GNN.



Source: Adapted from Hamilton (2020)

In addition to the different variants of spatial approaches, several general frameworks have been proposed to integrate different models into a single framework. One such framework is the message-passing neural network (MPNN), proposed by (Justin Gilmer et al., 2017). MPNNs are now well-established and use message-passing functions to unify several variants. Message passing is a fundamental operation in graph neural networks (GNNs). In message passing, each node in the graph sends a message to its neighbors, and the neighbors then update their own states based on the messages they receive. This process is repeated multiple times, and the final states of the nodes represent the graph's representation.

There are three steps for each node to generate embeddings during the neighborhood aggregation process:

1 **Receiving messages from its neighbors.** Every node in the graph computes a message for each neighbor. Messages are a function of the node, the neighbor, and the edge between them.

2 **Aggregating messages.** Every node aggregates the messages it receives using a permutation-invariant function (i.e., it doesn't matter in which order the messages are received). The aggregation function can be sum, average, max, or others.

3 **Updating its own features.** To encode the local structural information, each node updates its attributes (after receiving the messages) as a function of its current attributes and the aggregated messages.

Figure 6 depicts a single node aggregation messages from its local neighborhood. The model collects messages from the neighboring nodes (such as B, C, and D) from A's neigh-

borhood. These messages, in turn, are based on the information aggregated from the respective neighborhoods of B, C, and D. This process continues recursively, forming a tree structure in the computation graph of the GNN. The tree structure represents the unfolding of the neighborhood around the target node, highlighting the hierarchical nature of message aggregation in the GNN.

The MPNN framework, whose Equations 3.3 and 3.4 represent the *message* and *update* functions, respectively, is very similar to the general framework introduced in Algorithm 1:

$$m_u^k = \sum_{u \in \mathcal{N}(v)} \text{MESSAGE}_k(H_u^{k-1}, H_v^{k-1}, e_{uv}) \tag{3.3}$$

$$H_u^k = \text{UPDATE}_k(H_u^{k-1}, m_u^k) \tag{3.4}$$

In the Equation 3.4, $\text{MESSAGE}_k(,.,)$ defines the message between node u and v in the k-th layer, which depends on the two node representations u and v, as well as the information of their edge e_{uv}. The UPDATE_k is the node updating function in the k-th layer that combines the aggregated messages from the neighbors and the node representation itself. After K iterations of aggregation, the final node representation $\mathbf{H}_u^k$ in the last layer captures the structural information of the graph within K hops of distance in its neighborhood.

Moreover, the MESSAGE function in Equation 3.3 is the same as the AGGREGATE function in Algorithm 1 and performs a summation of all the messages from the neighbors. The node UPDATE function defined in Equation 3.4 is the same as the COMBINE function.

Finally, once the node representations have been learned by a GNN using the message-passing approach, the model can be applied directly to tasks such as node classification and link prediction (CHEN; TAO; WONG, 2021; DWIVEDI; BRESSON, 2021; DOU et al., 2020; LIU et al., 2021; KING; HUANG, 2023).

It is worth note GNNs are very accurate at link prediction tasks, which have been an active research topic (DIVAKARAN; MOHAN, 2020; DAUD et al., 2020; NASIRI et al., 2021), achieving state-of-the-art results on various datasets (ZHOU et al., 2020; WU et al., 2021). Notably, two strategies have proved successful for GNN-based link prediction. The first is to devise a score function that only depends on the two nodes that define a link. The second is to extract a subgraph around the focal link and solve link prediction by subgraph classification (ZHANG; CHEN, 2018b). In both cases, a GNN can learn node representations.

Unfortunately, existing GNNs face significant limitations. As GNN layers deepen, the representations of nodes in the graph tend to converge to a constant after sufficiently many layers, becoming indistinguishable from each other. This phenomenon, known as *over-smoothing* (CHEN et al., 2020; KELESIS et al., 2023; CHEN et al., 2020), severely limits the model's ability to learn distinct representations for nodes that belong to different classes. Another phenomenon is *over-squashing* (DIN; QURESHI, 2024), which occurs when a GNN struggles to compress information from exponentially growing neighborhoods into fixed-size node representations. This limits the model's ability to capture important long-range dependencies, especially in densely connected graphs. These limitations are particularly problematic in anomaly detection tasks, where subtle interactions across distant nodes must be captured to detect suspicious behavior patterns.

This inability to capture long-range dependencies significantly impacts performance, particularly in cybersecurity. Detecting sophisticated threats like advanced persistent threats (APTs) requires understanding global context and subtle, far-reaching interactions, a capability lacking in many existing GNNs. Addressing this limitation has driven research into adapting the Transformer architecture to graph data, a topic explored in the next chapter.

3.2 FROM TRANSFORMERS TO GRAPH TRANSFORMERS

The recent success of transformer architectures (VASWANI et al., 2017) in natural language processing and computer vision has inspired a surge of research into Graph Transformers (GTs). These models aim to leverage the strengths of transformers for modeling spatial relationships in graph-based tasks (LIU et al., 2023; WU et al., 2022; MA et al., 2023; MÜLLER et al., 2024). In fact, GTs are a relatively new type of GNN that leverages the attention mechanism. Due to its importance to this thesis, this chapter details transformer architecture, its constituent building blocks, and its relevance for graph-based machine learning tasks.

3.2.1 TRANSFORMER ARCHITECTURE

Transformers excel with sequential data, featuring a parallelizable architecture that effectively captures long-range dependencies. Undoubtedly, transformer architecture has been one of the most disruptive artificial intelligence technologies in recent years. The proliferation of newly introduced variants of transformer models has presented researchers with a formidable

challenge in terms of keeping up. First applied to the field of natural language processing for machine translation (VASWANI et al., 2017), the method is now a vibrant research topic in different domains thanks to its strong representation capabilities. Recent proposals include variants to computer vision (DOSOVITSKIY et al., 2021) to solve semantic segmentation (ZHENG et al., 2021), object detection (CARION et al., 2020), and super-resolution (ZHOU; LI; WANG, 2023); anomaly detection in time-series (KIM; KANG; KANG, 2023; Siva Kumar et al., 2023) and system logs (HUANG et al., 2020); and network intrusion detection (HAN et al., 2023b), to name a few. Moreover, generative AI has made substantial progress using transformer-based architectures (CHAVEZ et al., 2023; EKE, 2023).

The transformer architecture is composed of an encoder and a decoder. The primary function of the encoder is to generate encodings of the inputs, while the decoder leverages the assimilated contextual information from these encodings to produce the output sequence. Each transformer block encompasses various components: a multi-head attention layer, a feed-forward neural network, residual connections, and layer normalization, as depicted in Figure 7. The self-attention mechanism plays a central role in the architecture, through which the transformer learns the global dependency in a sequence.

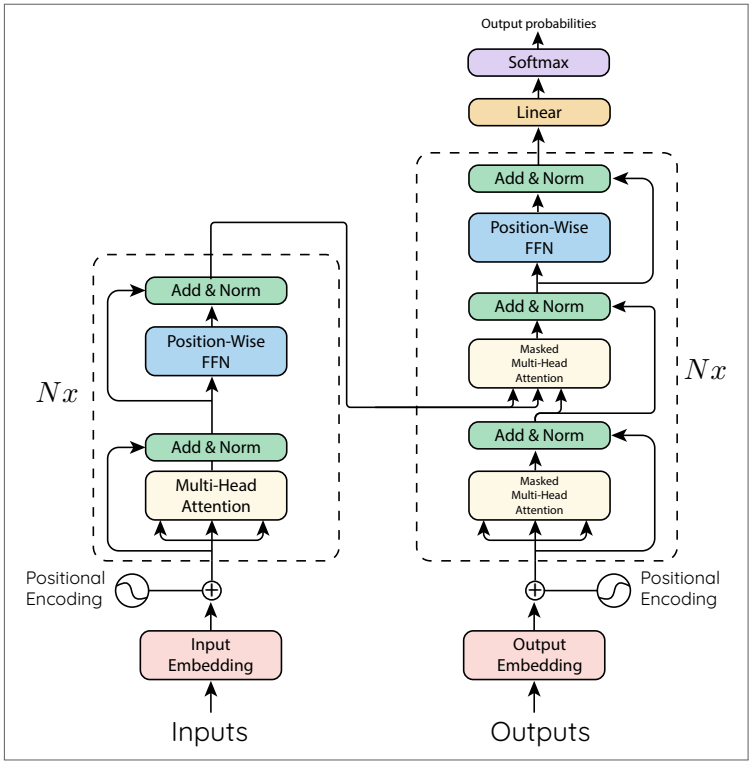
3.2.1.1 Self-attention

Self-attention is the basic building block of the transformer architecture, in both encoders and decoders. By employing the attention mechanism, the transformer model gains the ability to emphasize key segments within the input sequence, assigning varying weights to different parts accordingly. Specifically, this enables the model to selectively concentrate on essential elements during its processing by giving different weights to different parts.

Within the self-attention layer, the initial step involves transforming the input vector into three distinct vectors: the query vector $\mathbf{q}$, the key vector $\mathbf{k}$, and the value vector $\mathbf{v}$, all of which possess a dimensionality of $d_q = d_k = d_v = d_{\text{model}} = 512$. Subsequently, vectors derived from various inputs are organized and consolidated into three separate weight matrices: $\mathbf{Q}$, $\mathbf{K}$, and $\mathbf{V}$. These weight matrices are randomly initialized, and their weights are learned through training.

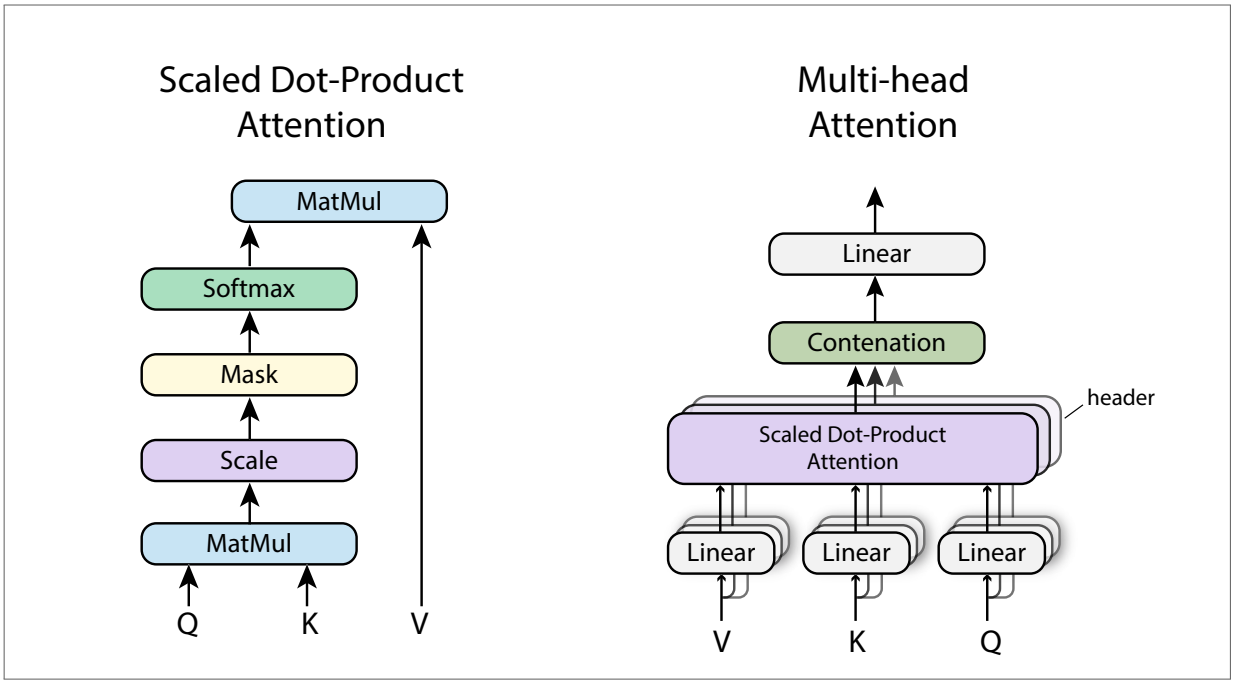
By calculating the self-attention values, the transformer model considers the entire sequence, enabling the capture of long-range dependencies and relationships between distant elements. The Equation 3.5 represents the calculation of attention values, while Figure 8(left

Figura 7 – The original transformer as introduced by Vaswani et al. (2017). The transformer architecture consists of an encoder-decoder structure with self-attention mechanisms. The encoder processes input sequences and captures contextual information, while the decoder generates output sequences using attention over the encoder's representations.



Source: original architecture from (VASWANI et al., 2017)

Figura 8 – The self-attention process expressed through Equation 3.5 and Multi-head attention helps the model expand the focus to various positions by providing different subspace representations. The figure is adapted from (VASWANI et al., 2017).



Source: adapted from (VASWANI et al., 2017)

side) depicts the process:

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d_k}}\right)\mathbf{V} \tag{3.5}$$

Specifically, the operation $\mathbf{Q}\mathbf{K}^\top$ computes scores between different input vectors, which determine the degree of attention that we give other input elements when encoding the element at the current position. To enhance gradient stability for improved training, the scores are then normalized after scaling by $\frac{1}{\sqrt{d_k}}$. The multiplication with matrix $\mathbf{V}$ yields a weighted value matrix, wherein the sum of the probabilities scales the value vector. Consequently, vectors associated with higher probabilities receive enhanced attention from subsequent layers.

Specific applications require a causal deep learning approach (OORD et al., 2016a; NICOLSON; PALIWAL, 2020). Specifically, when making predictions based on previous tokens, it is critical to prevent the attention mechanism from accessing any information about the token at future positions. In particular, the utilization of masking can guarantee the causality of the self-attention mechanism.

As pointed out by Nicolson e Paliwal (2020), to achieve this, a masking weight matrix $\mathbf{M}$ is used, where future positions are assigned a value of negative infinity ($-\infty$), and previous positions are assigned a value of 0. This masking operation is performed after scaling the multiplication of $\mathbf{Q}$ and $\mathbf{K}^\top$ by $\frac{1}{\sqrt{d_k}}$, and before the softmax operation in the self-attention calculation. The Equation 3.6 shows as the masked attention is computed:

$$\text{MaskedAttention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}\left(\mathbf{M} + \frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d_k}}\right)\mathbf{V} \tag{3.6}$$

By using $\mathbf{M}$ to mask out similarities $\mathbf{Q}\mathbf{K}^\top$ that include future positions, the new representation is computed so that softmax results in the actual scaled values for previous positions and the value 0 for future positions.

3.2.1.2 Multi-head attention

The multi-head attention mechanism provides different subspace representations, which helps the model expand the focus to various positions and capture diverse aspects of the same input. Instead of a single self-attention head, there can be h parallel self-attention heads, so the mechanism employs multiple queries with distinct weights to compute multiple attention values in parallel. These values are then concatenated to obtain a more comprehensive representation,

capturing richer information from multiple perspectives. The multi-head self-attention process is shown as follows:

$$\text{MHSA}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{contat}(head_0, ..., head_h)\mathbf{W}^0 \tag{3.7}$$

where $head_i = \text{Attention}(\mathbf{Q}\mathbf{W}_i^Q, \mathbf{K}\mathbf{W}_i^K, \mathbf{V}\mathbf{W}_i^V)$. In addition, $\mathbf{W}^0$, $\mathbf{W}_i^Q$, $\mathbf{W}_i^K$, and $\mathbf{W}_i^V$ are trainable parameters that can be learned through backpropagation during the training process.

The multi-head mechanism enables simultaneous attention functions across various representation subspaces of the input. As a result, each attention head can identify unique input characteristics, which are contained within different subspaces (LI et al., 2019a).

3.2.1.3 Positional Encoding

The transformer employs a technique known as positional encoding (PE) to incorporate the positional information of words in a sentence. Specifically, a positional encoding with dimension $d_{\textbf{model}}$ is added to the original input embedding. Let the input length be denoted by l, and the embedding dimension is given by d_{model}. Then, the positional encoding is a matrix of dimension $\mathbf{P} \in \mathbb{R}^{l \times d_{\text{model}}}$. Thus, every position in the input can be represented in terms of pos and the embedding depth d:

$$\text{PE}(pos, 2i) = sin\left(\frac{pos}{10000^{\frac{2i}{d_{\text{model}}}}}\right) \tag{3.8}$$

$$\text{PE}(pos, 2i+1) = cos\left(\frac{pos}{10000^{\frac{2i}{d_{\text{model}}}}}\right) \tag{3.9}$$

where pos denotes the position of words in a sentence, and i represents the current dimension of the positional encoding. In this way, each element of the positional encoding corresponds to a sinusoid, allowing the transformer model to learn to attend by relative positions and extrapolate to longer sequence lengths during inference. The function definition indicates that the frequencies decrease along the vector dimension and form a geometric progression from 2π to $10000 \cdot 2\pi$ in terms of wavelengths. Finally, the input representation $\mathbf{X}$ from transformer architecture is a composition from the word embedding $\mathbf{W}$ and positional encoding $\mathbf{P}$ matrices, which is represented by $\mathbf{X} = \mathbf{W} + \mathbf{P} \in \mathbb{R}^{l \times d_{\text{model}}}$.

Despite the role of positional encoding in incorporating word positions in the transformer, Haviv et al. (2022) argues that causal attention alone can approximate the absolute position of the inputs.

3.2.1.4 Positionwise Feed Forward Network

After the self-attention sub-layers in each encoder and decoder, a fully connected feed-forward network (FFN) is applied. For each position, linear transformation layers and a non-linear activation function within them are employed. The output of the feed-forward layer is calculated as follows:

$$\text{FFN}(\mathbf{X}) = \mathbf{W}_2\sigma(\mathbf{W}_1\mathbf{X}) \tag{3.10}$$

where $\mathbf{W}_1$ and $\mathbf{W}_2$ are the two weight matrices of the two linear transformation layers, and σ denotes the nonlinear activation function.

3.2.1.5 Residual connection (Add & Norm)

To achieve higher performance by strengthening the flow of information and to address the degradation of deep networks, a residual connection is added to each sub-layer in the encoder and decoder. A layer-normalization is followed after the residual connection to reduce the gradient dependencies between each layer, which speeds up the convergence as fewer iterations are needed (KAMATH; GRAHAM; EMARA, 2022; BA, 2016). In particular, the layer normalization ensures each layer has 0 mean and a unit variance. The output of these operations can be described as in Equation 3.11, where $\mathcal{F}$ represents operations like multi-head attention or feed-forward:

$$\text{AD}(\mathbf{X}) = LayerNorm(\mathcal{F}(\mathbf{X}) + \mathbf{X}) \tag{3.11}$$

3.2.1.6 The final layer of transformer

In the decoder, the *final layer* is responsible for converting the stack of vectors back into a word. This process involves two steps: a linear layer and a softmax layer. The linear layer

projects the vector into a logit vector whose dimensions correspond to several words in the vocabulary. Subsequently, the softmax layer is applied to the logits vector. This layer transforms the logits into probabilities, assigning a probability value to each word in the vocabulary. As pointed out by Han et al. (2023a), these probabilities indicate the likelihood of each word being the correct output word.

3.2.2 TRANSFORMER FOR GRAPH REPRESENTATION LEARNING

Transformer architecture for graphs is a recent and promising research topic that has gained growing attention in graph representation learning literature(CHEN; TAO; WONG, 2021; CHEN; O'BRAY; BORGWARDT, 2022; YING et al., 2021; SHI et al., 2021; HU et al., 2020; MÜLLER et al., 2024).

Some authors argue that even pure transformers are already powerful graph learners Kim et al. (2022), Henderson et al. (2023). This is because the self-attention mechanism in the standard Transformer considers the input tokens as a fully connected graph, which is agnostic to the inherent graph structure of the data. Notably, one of the key advantages of Transformer variants for graph modeling (CHEN; TAO; WONG, 2021; DWIVEDI; BRESSON, 2021) is that they can learn long-range dependencies between nodes. The self-attention mechanism allows the model to attend to any pair of nodes in the graph, regardless of their distance, and each attention head implicitly attends to information from a different representation subspace of different nodes. This allows the model to learn different aspects of the node representations. As previously pointed out in Chapter 1, capturing global context and detecting subtle, long-range interactions is critical for identifying anomalous behavior in cybersecurity contexts. Unlike traditional GNNs, which only aggregate local neighborhood information, Transformers can capture interaction information between any node pair via a single self-attention layer.

The superiority of GTs over GNNs lies in their capacity to capture long-range dependencies and global information by leveraging the attention mechanism, which GNNs, usually limited to local structure, often fail to grasp. As noted by Müller et al. (2024) and Chen, O'Bray e Borgwardt (2022), the global attention mechanism allows GTs to capture relationships between any two nodes, regardless of their distance, overcoming the *over-smoothing* and limited expressiveness often seen in deep GNNs. By considering all nodes during information aggregation, GTs are also less susceptible to *over-squashing*. This enables GTs to preserve information flow better and capture long-range dependencies in complex graphs.

According to authors in (YING et al., 2021), the key to utilizing Transformer in graphs is to properly incorporate structural information of graphs into the model, such as structural relation between nodes and node importance. The authors propose using the shortest path distance between any two nodes to model spatial relations and accurately capture the spatial dependency in a graph. The authors propose to encode the node centrality to capture the node importance. In turn, Chen, Tao e Wong (2021) explore transformers to graphs by projecting an efficient node sampling method to reduce the time and space complexity of the attention mechanism, as well as modifications on the attention mechanism to learn diverse node relations on graph topology.

In their paper, Shi et al. (2021) employs a Graph Transformer network as a feature propagation component in a hybrid architecture that unifies feature propagation and label propagation. This architecture is empirically robust on the semi-supervised node classification task, and they obtain new state-of-the-art semi-supervised classification results.

Despite their strengths, GTs are not without challenges. The primary drawback is the quadratic complexity of the attention mechanism, which requires computing pairwise attention scores between all nodes. As discussed by recent literature (WU et al., 2022; CHEN; TAO; WONG, 2021; WU et al., 2023), this computational overhead limits the scalability of GTs, particularly for large graphs commonly encountered in real-world applications like cybersecurity.

To address the scalability issue, several approaches have been proposed to improve the efficiency of Graph Transformers by reducing the computational complexity of the attention mechanism. For example, Node Sampling involves (CHEN; TAO; WONG, 2021) selecting a subset of nodes to participate in the self-attention mechanism. This reduces the number of tokens considered during attention calculation, making the process more efficient.

The authors in (WU et al., 2022) proposed a scalable and efficient graph Transformer for node classification that can propagate layer-wise node signals between arbitrary node pairs through a kernelized Gumbel-Softmax operator with linear algorithmic complexity, which can distill latent structures among all the instance nodes. In (CHEN; O'BRAY; BORGWARDT, 2022), the authors have presented a new type of flexible Graph Transformer that uses a structure-aware self-attention mechanism. This mechanism has achieved state-of-the-art performance on graph and node classification tasks by incorporating structural information through a subgraph extraction rooted at each node before computing the attention.

Hybrid Approaches allow us to combine different efficient transformer techniques, which can lead to further efficiency gains. For instance, combining global attention with local message-

passing has shown promise, as seen in GraphGPS (RAMPášEK et al., 2022) models. The research proposed by Wu et al. (2023) proposes a Simplified Graph Transformer (SGFormer) that employs a single-layer global attention mechanism that scales linearly w.r.t. the number of nodes, significantly reducing computational complexity and memory usage.

The work in (LIU et al., 2023) introduces Gapformer for node classification. Gapformer incorporates Graph Pooling to overcome existing limitations of transformers with large graphs. The method coarsens the large graph into a smaller set of “pooling nodes"using local or global graph pooling strategies. This allows for sparse attention mechanisms, which focus only on the pooling nodes rather than all other nodes, resulting in linear complexity and reduced noise.

To address the computational burden of Graph Transformers, the research in (LIU et al., 2024) delves into sparsification techniques to reduce their complexity. The authors tackle this challenge by identifying and removing redundant computations within GT models. They achieve this through pruning strategies applied to various components, including input nodes, attention heads, model layers, and model weights. While these pruning methods offer the potential for efficiency and generalizability, several challenges remain. For instance, dynamically adjusting the pruning ratio for different graph structures and effectively pruning all components while accounting for their intricate interactions presents significant hurdles.

Each method aims to reduce the cost of attention in Graph Transformers while maintaining the model's performance. However, despite these advancements, the adoption of GTs for cybersecurity anomaly detection remains limited despite their potential benefits. This stands in contrast to the wider use of Graph Neural Networks (GNNs) in this domain. Moreover, the related works mentioned above suggest that academic research involving applications or generalizations of the Transformer architecture on graph data has been a very active and rapidly growing research topic.

3.3 LINK PREDICTION THROUGH GRAPH NEURAL NETWORKS

Since real-world graphs are often only partially observed, the central problem is to predict missing links, which have been an active research topic (DIVAKARAN; MOHAN, 2020; DAUD et al., 2020; NASIRI et al., 2021). In particular, link prediction aims to forecast the likelihood of a connection between two nodes within a graph. In this work, we reframe the problem of detecting unusual authentications between entities across a computer network as the problem of predicting low-probability links in a graph.

GNNs are very accurate at link prediction, achieving state-of-the-art results on various datasets (ZHOU et al., 2020; WU et al., 2021). Notably, two strategies have proved successful for GNN-based link prediction. The first is to devise a score function that only depends on the two nodes that define a link (KIPF; WELLING, 2016; KOLLIAS et al., 2022; ZHU et al., 2023). The second is to extract a subgraph around the focal link and solve link prediction by subgraph classification (ZHANG; CHEN, 2018b). In both cases, a GNN can learn node representations.

A simple score function, such as inner product, cosine similarity, or even parameterized functions like bilinear, typically operates only on the embeddings of the two target nodes. This inherently limits the amount of relational and structural information that can be captured. Such functions may struggle to disambiguate links in complex graph topologies. In contrast, a link classifier is designed to process richer contextual information. By operating on the subgraph surrounding the link, the classifier can learn to capture structural patterns, such as position-aware features that are inaccessible to node-pair-only scoring mechanisms. Empirical studies, such as those by Zhang e Chen (2018b), demonstrate that subgraph-based link classifiers significantly outperform simple score functions in a variety of benchmarks, especially when the link prediction task involves complex relationships or requires robustness to noise. Thus, in our experiments, we adopt subgraph classification for the link prediction strategy, in which a GNN is trained jointly with the link classifier.

Especially, we extend the vanilla multi-head attention (VASWANI et al., 2017) into the graph representation learning problem, where the input is an authentication graph. By modeling the problem through the lens of GNNs, we can incorporate the behavioral characteristics of users and their interactions within the network. It is worth noting that such is more challenging than using traditional machine learning for classification, which requires the information to be independent. Specifically, we want to represent a computer network as a graph-structured dataset, where users and machines are nodes, and authentications are denoted by edges. The idea is that low-probability edges reflect atypical logons and, by extension, possible unauthorized access we want to detect. It can be a challenge as computer networks are dynamic graphs, meaning that the nodes and edges in the graph can be added or removed over time. This can make it difficult for GNNs to learn a generalizable model for future data.

To overcome this challenge, in Chapter 4, this work investigates the Transformer architecture to identify abnormal authentications or suspicious hosts within computer networks, which may signal lateral movement. Specifically, we redefine the problem of detecting atypical logins as a graph link prediction task. Since real-world graphs are often only partially observed,

predicting missing links becomes a critical challenge.

In Chapter 5, we introduce a new Graph Transformer model designed to detect anomalous graph nodes through a node classification task. This model is particularly useful for identifying suspicious hosts within a network, allowing us to isolate compromised endpoints even during non-APT attacks. We propose that these approaches are complementary, as the models address the intrusion detection problem from two distinct but supportive perspectives.

3.4 CHALLENGES IN USING GNN FOR GRAPH REPRESENTATION LEARNING IN THE CYBER SECURITY CONTEXT

Notably, GNN models include Graph Convolutional Networks (GCN) (KIPF; WELLING, 2017), GraphSAGE (HAMILTON; YING; LESKOVEC, 2017), Graph Attention Networks (GAT) (CASANOVA; LIO; BENGIO, 2018), and Graph Isomorphism Network (GIN) (XU et al., 2019). For example, while GCN is popular and praised for its simplicity, GraphSAGE uses fixed-neighborhood sampling to save computation and suggests different aggregators to optimize the representation: mean aggregator, LSTM aggregator, and pooling aggregator, where mean aggregator can be regarded as an inductive version of GCNs. The LSTM aggregator is not permutation invariant, which means that the order of the sampled neighbors matters. This can be a problem for graphs where the order of the nodes is not meaningful.

The authors of GIN (XU et al., 2019) identified graph structures that GNNs, such as GCNs and GraphSAGE, cannot distinguish. They then proposed their method as a more expressive GNN architecture, which achieves the same representational power as the Weisfeiler-Leman (WL) test (LEMAN; WEISFEILER, 1968), an algorithm for graph isomorphism testing.

Meanwhile, the work in (BRESSON; LAURENT, 2018) proposed a generic graph ConvNet design that uses gated edge mechanisms inspired by LSTM and residuality. Using a simplified design, the proposed method achieved promising performance on subgraph matching and graph clustering tasks.

GATs are a powerful and flexible GNN architecture that learns the importance of each node's neighbor based on attention mechanisms during the propagation step. This allows GATs to compute each node's hidden states by attending to its neighbors.

Due to its convincing performance, GNN has become an important ally in the cybersecurity domain (LAKHA et al., 2022; KING; HUANG, 2023; NGUYEN et al., 2022; SUN; YANG, 2022; ZHOU et al., 2019b). For example, the authors in (SUN; YANG, 2022) and (KING; HUANG, 2023) explored

graph neural networks to detect lateral movement by discovering anomalous links between nodes on a graph that represent anomalous authentications.

However, the success of detecting anomalous authentications using GNNs is conditioned on the representational power and performance of those models. The combination of the representations of a node and its neighbors, which we call aggregation, contributes significantly to the representational power of a GNN. In general, graph neural networks aim to learn permutation invariant hypotheses to have consistent predictions for the same graph when presented with permuted vertices/nodes, and such property is achieved through neighborhood aggregation schemes. The GNN mentioned above models traditionally use simple functions (e.g., sum, max, mean) on the node embeddings to preserve permutation invariance. However, we argue that an effective aggregation of node features into a graph-level representation can not be achieved through simple sum or mean. Moreover, a single outlier can easily distort aggregation functions such as sum, weighted mean, or the max operation used in standard GNNs, leading to inaccurate or biased results.

In Chapter 4 from this thesis, we adopt the subgraph classification approach to link prediction proposed in SEAL (ZHANG; CHEN, 2018b). SEAL adds a Double-Radius Node Labeling (DRNL) strategy to labeling node attributes (i.e., node's structural label), where the purpose is to use different labels to mark different roles of nodes. Afterward, the method generates node embeddings using a Deep Graph Convolutional Neural Network (DGCNN) (ZHANG et al., 2018). In contrast to SEAL, we utilize multi-head self-attention to generate node embeddings, inspired by recent advances in adapting the transformer architecture for learning representations in graph-structured data, as discussed in Subsection 3.2.2 of this chapter.

Learning through Graph Neural Networks (GNNs) can be challenging because of the possibility of aggregating misleading information from the neighborhood during message passing. This can lead to suboptimal performance since these misleading messages can spread throughout the entire graph as GNNs go deeper, ultimately affecting the overall outcome. Thus, it is essential to filter out irrelevant information to ensure optimal performance. For example, in a recent study (CHEN et al., 2021), the authors propose an approach that explicitly prunes the irrelevant neighbors in the message-passing stage of GCNs, significantly reducing the negative impact of noise in recommender systems. On the contrary, within the scope of our research, rather than filtering pertinent information by eliminating neighboring vertices, we delve into a methodology that emphasizes relationships of higher relevance within a graph, utilizing a residual soft-attention aggregation methodology. As our results demonstrate, such an appro-

ach proved to be pivotal in our work, enhancing the performance of our model compared to models that employ simpler aggregation functions.

Moreover, GNNs have been demonstrated to be inherently susceptible to the problems of over-smoothing (KERIVEN, 2022; NGUYEN et al., 2023), in which nodes tend to have similar representations after the aggregation operation as models become deeper. While increasing GNN depth can capture long-range dependencies and eventually improve its representational power, it is well-known that deeper GNN models do not always enhance performance and may even deteriorate it (ZHOU et al., 2020). This is primarily attributed to the fact that additional layers can propagate noisy information from an exponentially growing number of expanded neighborhood members.

The recent work (CHEN et al., 2022) proposed overcomes these limitations in the context of recommendation systems by presenting a new approach to capture long-range dependencies without increasing the depths of GNNs. In particular, the authors overcome the problem by grouping similar nodes using K-Means and connecting individual nodes to their corresponding centroids by computing node-centroid attentions. This enables long-range information flow via non-local attention among distant but similar nodes. This "bridging"mechanism jointly learns GNNs and k-Means in one unified model. It lets shallow GNNs capture both local and non-local relationships, avoiding the bottleneck effects of deep GNNs. However, using the K-Means clustering algorithm can impose an extra computational load.

Additionally, the model's performance depends heavily on selecting the correct cluster size, which is a critical hyperparameter of K-Means. When a sub-optimal cluster size is selected, the success rates of those approaches become limited. Conversely, our method is more straightforward and does not rely on additional optimizations.

There is another critical challenge related to spatial approaches, as previously mentioned, which define convolutions directly on the graph based on its topology. This challenge involves defining the convolution operation with differently sized neighborhoods while maintaining local invariance on the graph topology. This problem is yet to be solved and can offer significant scope for academic contributions.

4 DETECTING ABNORMAL LOGINS BY DISCOVERING ANOMALOUS LINKS VIA GRAPH TRANSFORMERS

Detecting anomalous authentications is essential for mitigating cyber attacks, particularly APTs, where gaining unauthorized access is fundamental to the attacker's strategy in such attacks. As previously discussed, the APTs represent a severe and furtive threat to cyberinfrastructure and are executed over prolonged periods. Their low and slow approach often characterizes these attacks, making them notoriously difficult to detect and counter. The ability to detect and respond to these anomalous authentication attempts is a technical challenge necessary for society's cyber resilience.

This chapter approaches a new Transformer-inspired GNN model to detect abnormal logins. Especially, the proposed model introduces a new residual soft-attention scheme that facilitates the aggregation of node representations through a weighted sum, resulting in enhanced node representations and improved filtration of irrelevant information.

4.1 THE ROLE OF ABNORMAL AUTHENTICATIONS IN ADVANCED CYBER THREATS

As stated by authors in (ALSHAMRANI et al., 2019), during an APT, the attackers will often keep stealthy and slowly expand their presence by moving laterally within the network once they have gained access, searching for sensitive data or critical systems. One of the key ways that APTs can move laterally is by using anomalous authentications. This means that they will attempt to obtain access to systems through unauthorized logins using credentials that do not belong to them. This can be done by stealing credentials, using brute force attacks, or exploiting vulnerabilities in the authentication process. Figure 2 from Chapter 2 depicts the process.

As previously noted in that figure, machines *A*, *B*, *C*, *D*, *E*, and *F*, as well as servers *SRV-1* and *SRV-2*, can be represented as nodes in a graph, and authentication events between these entities in the network can be represented as edges in the graph. If a user from machine C does not usually have access to server *SRV-2*, an attacker can gain legitimate access to it. An accurate graph machine learning model could identify this access as anomalous.

We focused on detecting abnormal authentications, as they can reflect cyber threats such as lateral movement, in which a user typically gains unauthorized access to different machines to consolidate their presence in the target network. Therefore, detecting abnormal authentications

is essential to prevent or mitigate lateral movement and, consequently, to identify an ongoing APT attack within an organization. However, traditional security products often fail to detect lateral movement because of the dynamic nature of the attack. Therefore, solutions that adapt to the changing behavior of attackers are needed.

Fortunately, attackers leave traces in the form of authentication logs during unauthorized access, which we can use to evaluate the performance of machine-learning-based detection models. Crucially, these logs tell us which users and machines were involved in the recorded events. Moreover, such resources are essentially graph-structured data and intrinsically relational. In these data structures, the users and machines are represented by nodes, whereas the edges denote the authentication events between them. Therefore, graph representation learning has the potential to make a significant difference over traditional machine learning in cyber security research (BOWMAN; HUANG, 2021).

While other approaches can use graph neural networks to solve the problem of identifying atypical authentications (SUN; YANG, 2022; KING; HUANG, 2023), the incorporation of the multi-head attention mechanism derived from transformers represents an evolution in the way models gather information on the graph and help to identify these abnormal behaviors across the computer network. It occurs because the multi-head self-attention obtains a more comprehensive representation, capturing richer information regarding user behavior within the network from multiple perspectives. As mentioned earlier in Chapter **??**, Section 3.2.1, the multi-head attention mechanism can simultaneously extract various types of information from several representation subspaces, capturing distinct input properties.

In particular, unlike traditional GNNs, the large contextual range of the transformers enables the aggregation of information from arbitrary nodes, which overcomes the shortcomings of the GNNs in long-term dependencies.

As we will describe in greater detail later in this chapter, we evaluated the performance of our method on three cybersecurity-related datasets, which are particularly useful for testing models for detecting anomalous authentications. Specifically, we evaluated its performance concerning the AUC, TPR, FPR, and F1 score metrics. The approach proposed in this chapter was recently published as a full paper in the *Computers & Security Journal* (GONçALVES; ZANCHETTIN, 2024a).

4.2 AN OVERVIEW OF THE PROPOSAL FOR DETECTING ABNORMAL LOGINS

The concept underlying our proposal, elaborated in Section 4.3, is that users compromised in a lateral movement attack will engage with devices they typically cannot access. As such, the model profiles each network entity and its relationships to neighboring entities, identifying authentication activities diverging from standard patterns. However, the success of detecting anomalous authentications using GNNs is conditioned on the representational power and performance of those models. The combination of the representations of a node and its neighbors, which we call aggregation, contributes significantly to the representational power of a GNN. In general, graph neural networks aim to learn permutation invariant hypotheses to have consistent predictions for the same graph when presented with permuted vertices/nodes, and such property is achieved through neighborhood aggregation schemes. Graph neural networks traditionally use simple functions (e.g., sum, max, mean) on the node embeddings to preserve permutation invariance. However, we argue that an effective aggregation of node features into a graph-level representation cannot be achieved through a simple sum or mean. These aggregation techniques fail to filter out redundant information and thus propagate uninformative features.

To overcome that problem, we aggregate the node representations during message passing through a residual soft attention mechanism that transforms the node representations and then uses a weighted sum with gating vectors to aggregate across nodes. This scheme enhances the aggregation of node representations by the graph transformer, selecting relevant node representations through soft attention and consequently reducing the influence of redundant information.

Our choice of attention-based node aggregation is inspired by previous works that used similar approaches to solve tasks such as program verification (LI et al., 2016) and graph similarity learning (LI et al., 2019b). These approaches are related to soft alignment and attentional (also known as global attention) models (SUKHBAATAR et al., 2015; HE; WU; LI, 2021) in that they use context to focus attention on the nodes that are most important to the current decision. This is important for the task we are working on, as we need to be able to identify the most relevant nodes in the graph to make a good decision. In soft attention, a probability distribution is calculated over all positions in the input. The resulting probabilities reflect each position's relative importance and are used as weights to produce a context vector. Soft attention is fully differentiable, so it can be easily trained using gradient-based methods.

Moreover, complementary to attention coefficients in multi-head attention is the idea that the model should be aware of the contribution of each edge at the neighborhood from a node. For this reason, we incorporate that idea in our model by employing a gating mechanism so that the model can decide which edges are more relevant for the task. The introduced gating mechanism helps the model focus on more significant relationships on the graph by determining the relative importance of edges during a message-passing event.

Drawing inspiration from LSTM (HOCHREITER; SCHMIDHUBER, 1997) concepts to effectively model syntactic dependency graphs, the authors in (MARCHEGGIANI; TITOV, 2017) notably explored the introduction of gates in GCNs. Their work, in turn, was inspired by (OORD et al., 2016b) and (DAUPHIN et al., 2017), which previously studied the incorporation of gating mechanism in convolutional neural networks, respectively, regarding the conditional image generation and language modeling tasks.

In the present approach, we argue that graph transformers can benefit from such a strategy during message passing by considering the most critical edges. We highlight the main contributions from this approach as follows:

- We introduce a residual soft-attention scheme for graph transformer networks that facilitates the aggregation of node representations through a weighted sum, resulting in enhanced node representations and improved filtration of irrelevant information.
- We incorporate a gating mechanism that aids the model in determining the relative importance of edges during the message-passing process.
- We emphasize reducing the False Positive Rate (FPR). In cybersecurity, a high FPR can lead to many false alarms, consuming unnecessary resources and potentially overlooking real threats. The proposed model demonstrates its real-world applicability by focusing on this metric and maintaining it below an industry-accepted threshold.

The introduction of the residual soft-attention aggregation scheme, which decides which nodes are relevant to the current graph-level task, and the gating mechanism differentiates this research from conventional methods. Moreover, leveraging the multi-head attention mechanism from transformers can be seen as a contemporary approach to solving the problem.

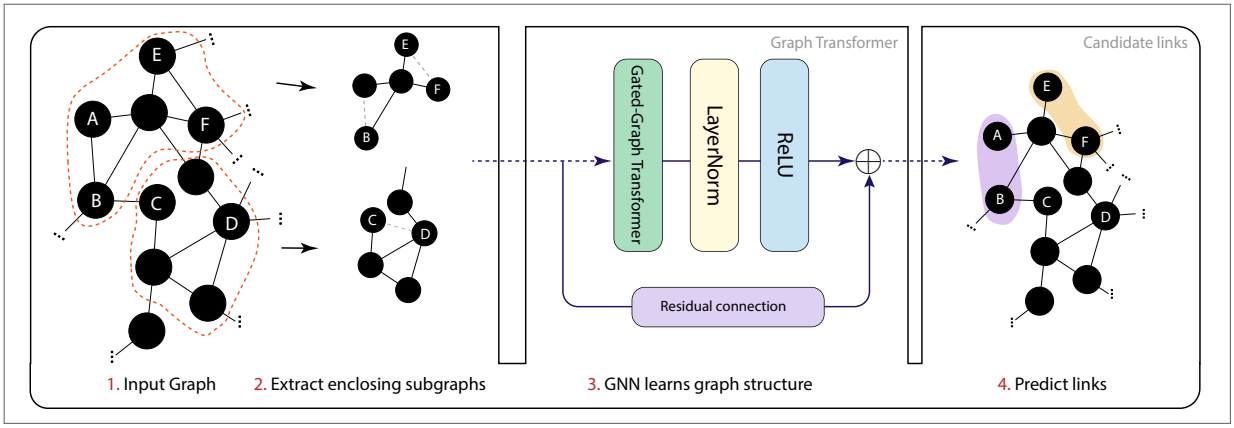
Through extensive experimentation, we demonstrate the feasibility of the proposed methodology. We evaluated our model's performance using multiple metrics to understand how

it works comprehensively. The experimental results show that the proposed method outperforms state-of-the-art approaches such as GAT (CASANOVA; LIO; BENGIO, 2018), GCN (KIPF; WELLING, 2017), and SAGE (HAMILTON; YING; LESKOVEC, 2017) in fundamental evaluation metrics, including AUC, TPR, and, crucially, FPR. FPR is a critical metric in cybersecurity, as a model's ability to maintain a low false positive rate is essential for its real-world applicability in detecting cyber threats. Even more important, the lower the false positive rate, the lower the investigation budget.

4.3 GATED-GRAPH TRANSFORMER MODEL

Inspired by work in (ZHANG; CHEN, 2018b), we have extracted the enclosing subgraphs surrounding node pairs of interest, namely, the focal nodes, and used these subgraphs to build a node feature matrix. Specifically, this matrix encapsulates the structural variances of the vertices situated at varying distances from each focal pair. The idea is that structural differences help the GNN to predict the existence of links between the nodes. As detailed in the following sections, we have modified a Graph Transformer Network by integrating novel features, thereby empowering it to generate enhanced representations of nodes for link prediction. The proposed approach is referred to as *Gated-Graph Transformer*.

Figura 9 – Illustration of how our model learns general graph structure features from local enclosing subgraphs using a transformer-based propagation module that transforms the node representations through a soft attention mechanism and then uses a weighted sum with gating vectors to aggregate across nodes while helping filter out irrelevant information. In addition, we have implemented a residual connection to prevent over-smoothing



Source: Thesis author

Specifically, we use $\mathcal{G}^h(u, v)$ to denote the h-hops neighborhood subgraph of the node pair (u, v), which is derived from the whole graph $\mathcal{G}$. For any node v_k in the neighborhood subgraph

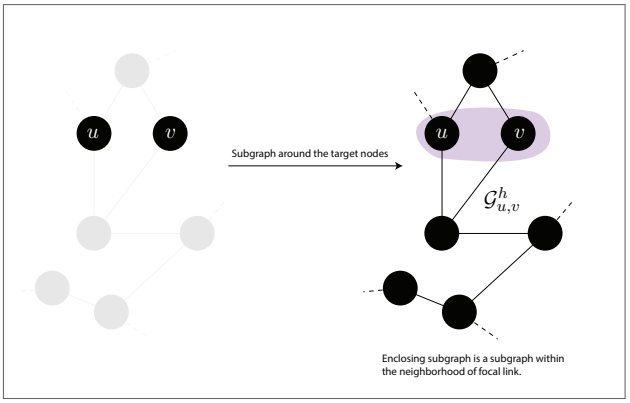
$\mathcal{G}^h(u,v)$, it should satisfy $d(u,v) \leq h$, indicating that $v_k \in \mathcal{N}^h(u) \cup \mathcal{N}^h(v)$. Figure 9 depicts the proposed methodology.

4.3.1 TAKING GENERAL GRAPH STRUCTURE FEATURES FOR LINK PREDICTION

Before training the model, we build structural node features by extracting subgraphs around the focal links and solve link prediction by (sub)graph classification. However, unlike in vanilla graph classification, where a priori all links are equally important, in graph classification for link prediction, the focal link plays a special role and the relative positions of other links concerning it matter. In particular, our method to compute structural features for link prediction follows the same criteria from (ZHANG; CHEN, 2018b).

Specifically, we use local subgraphs to compute structural features as those structures include rich information related to link existence and already contain enough information to learn good graph structure features for link prediction. Thus, we extract local enclosing subgraphs around focal links as training data and use the GNN to learn which subgraphs correspond to link existence. Concretely, the enclosing subgraph $\mathcal{G}^h_{u,v}$ for a node pair (u,v) is the subgraph induced from the network $\mathcal{G}$ by the union of u and v's neighbors up to h-hops, as depicted in Figure 10.

Figura 10 – The enclosing subgraph $\mathcal{G}^h_{u,v}$ for a focal node pair (u,v) is the subgraph induced from the network $\mathcal{G}$ by the union of u and v's neighbors up to h hops.



Source: Thesis author

Within an enclosing subgraph, nodes with different relative positions to the center $(u,\ v)$ have different structural importance to the link. To mark those different roles of the nodes, we use distinct labels to indicate *where are the target nodes between which a link existence*

should be predicted. In particular, we let nodes with the same distance to the central node have the same label so that the node labels can reflect nodes' relative positions and structural importance within subgraphs.

Let u and v be the two central nodes. Intuitively, the topological position of a node i within an surrounding subgraph can be described by its distances (or its radius) to the two central nodes, specifically $(d(i,u), d(i,v))$, where where d denotes the shortest path distance between two nodes. Thus, we assign the same label to nodes that share the same orbit, allowing these labels to reflect the relative positions and structural importance of the nodes within the subgraph. Based on this criteria, Zhang e Chen (2018b) propose a DRNL function, which iteratively assigns larger labels to nodes that have a greater radius with respect to both central nodes.

After computing the structural label of each node in an enclosing subgraph, we then use the one-hot encoding vectors from those labels to construct a node feature matrix X, which is used to train the GNN model.

We summarize the above steps as follows:

1. **Enclosing subgraph extraction.** The center nodes u and v are the target nodes between which the link is located. The enclosing subgraph $\mathcal{G}^h_{u,v}$ for a focal node pair (u,v) is the subgraph induced from the network $\mathcal{G}$ by the union of u and v's neighbors up to h hops. Such subgraph contains all the information in the neighborhood of node u, v.

2. **Node information matrix construction.** In an enclosing, nodes that are positioned differently relative to the central nodes have varying structural significance for the link and receive different labels accordingly. Therefore, we compute their structural label and then use one-hot encoding vectors from these labels to construct a node feature matrix X.

3. **GNN training.** The node feature matrix and the subgraphs are utilized to train the GNN. In the link prediction task, we treat it as a subgraph classification problem where a Multi-Layer Perceptron (MLP) classifier predicts the link existence between pairs of nodes.

4.3.2 GETTING NODES REPRESENTATIONS

We adapted vanilla transformers to generate node embeddings. Even pure transformers are already powerful graph learners (KIM et al., 2022). This is because the self-attention mechanism in the standard Transformer considers the input tokens as a fully connected graph, which is agnostic to the inherent graph structure of the data. Notably, one of the key advantages of Transformer variants for graph modeling is that they are able to learn long-range dependencies between nodes. This is because the self-attention mechanism allows the model to attend to any pair of nodes in the graph, regardless of their distance, and each attention head implicitly attends to information from a different representation subspace of different nodes. This allows the model to learn different aspects of the node representations.

Specifically, given node features $H^{(l)} = \{h_1^{(l)}, h_2^{(l)}, ..., h_n^{(l)}\}$, we calculate multi-head attention coefficient for each edge as in Equation 4.1:

$$\begin{aligned} q_{c,u}^{(l)} &= \mathbf{W}_{c,q}^{(l)} h_u^{(l)} + b_{c,q}^{(l)} \\ k_{c,v}^{(l)} &= \mathbf{W}_{c,k}^{(l)} h_v^{(l)} + b_{c,k}^{(l)} \\ \alpha_{c,uv}^{(l)} &= \frac{\exp\left(\frac{{q_{c,u}^{(l)}}^\top k_{c,v}^{(l)}}{\sqrt{d}}\right)}{\sum_{\zeta \in \mathcal{N}(u)} \exp\left(\frac{{q_{c,u}^{(l)}}^\top k_{c,\zeta}^{(l)}}{\sqrt{d}}\right)} \end{aligned} \tag{4.1}$$

where $\exp\left(\frac{q^\top k}{\sqrt{d}}\right)$ is exponential scale dot-product function and d is the hidden size of each head. For the c-th head attention, we firstly transform the features $h_u^{(l)}$ and $h_v^{(l)}$ from nodes (u, v) into query vector $q_{c,u}^{(l)} \in \mathbb{R}^d$ and the key vector $k_{c,v}^{(l)} \in \mathbb{R}^d$ respectively using different trainable parameters $\mathbf{W}_{c,q}^{(l)}$, $\mathbf{W}_{c,k}^{(l)}$, $b_{c,q}^{(l)}$, $b_{c,k}^{(l)}$.

Additionally, we introduce a gating mechanism to help the model learn what edges are important for the graph learning task at hand, which is expressed by Equation 4.2:

$$\eta_{u,v} = \sigma(\mathbf{W}_u^{(l)} \mathbf{h}_u^{(l)} + \mathbf{W}_v^{(l)} \mathbf{h}_v^{(l)}) \tag{4.2}$$

where σ is the logistic sigmoid function, and $\mathbf{W}_u^{(l)}$ and $\mathbf{W}_v^{(l)}$ are learnable weight for the gate. The sigmoid function $\sigma(\cdot)$ from the Equation 4.2 plays a critical role in the gating mechanism by enabling the model to learn a differentiable importance score for each edge (u, v) into a node's neighborhood. More specifically, this gating mechanism is designed to regulate the flow of information between node pairs during message passing by learning a

continuous attention weight over the graph edges. Specifically, the sigmoid function maps its input to the interval $(0, 1)$, making it particularly well-suited for expressing attention-like weights or gating values that represent relative importance. In this formulation, $\eta_{u,v}$ can be interpreted as a soft indicator of how important the edge (u, v) is for the current learning task.

As our model employs multi-head attention over all nodes, after getting the graph multi-head attention, we proceed to node update as follows in Equation 4.3:

$$\hat{h}_u^{(l+1)} = \Big\Vert_{c=1}^{C} \Big[\sum_{v \in \mathcal{N}(u)} (\alpha_{c,uv}^{(l)} \mathbf{W}_{c,v}^{(l)} h_v^{(l)}) \odot \eta_{c,uv} \Big] \tag{4.3}$$

where $\|$ is the concatenation operation for C head attention and $h_v^{(l)}$ is the v-th node's features at the l-th layer.

To increase model stability, existing works that explore the transform architecture in graph representation learning introduce gated residual connections between layers. An important example can be found in (SHI et al., 2021), and those residual connections can be mathematically denoted as in Equation 4.4:

$$\begin{aligned} r_u^{(l)} &= \mathbf{W}_r^{(l)} h_u^{(l)} + b_r^{(l)} \\ \beta_u^{(l)} &= \sigma(\mathbf{W}_{\text{gate}}^{(l)} [\hat{h}^{(l+1)}; r_u^{(l)}; \hat{h}_u^{(l+1)} - r_u^{(l)}]) \\ h_u^{(l+1)} &= \text{ReLU}(\text{LayerNorm}(1 - \beta_u^{(l)}) \hat{h}_u^{l+1} + \beta_u^{(l)} r_u^{(l)}) \end{aligned} \tag{4.4}$$

where σ denotes the sigmoid. In our experiments, such a strategy showed a significant gain in model quality.

4.3.3 AGGREGATION OF NODE FEATURES

As previously discussed in Chapter 3, the GNNs combine the feature information of the nodes and the graph structure to learn better representations of the graph. The general idea is that GNNs use a form of neural message passing in which vector messages are exchanged between nodes and updated using neural networks (Justin Gilmer et al., 2017). In neural message passing, each node in the graph sends a message to its neighbors. As depicted in Figure 6, these messages are aggregated and updated to the node's representation. In each iteration of a GNN, the node embeddings $\mathbf{h}_u^{(l)}$ of each node u in the graph $\mathcal{V}$ are updated by aggregating information from its neighbors $\mathcal{N}(u)$.

The message-passing update can be interpreted as propagating information through the graph. In each iteration, the node embeddings are updated to reflect the information propagated from their neighbors. This process continues for several iterations until the node embeddings reflect the overall structure of the graph.

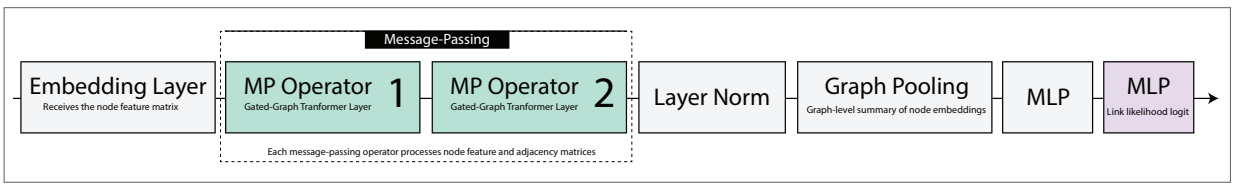
In this work, we introduce a residual soft attention mechanism on the Graph Transformer Network while aggregating messages from neighbor nodes that transform the node representations. Then, we use a weighted sum with gating vectors to aggregate across nodes. Our aggregation function is mathematically represented as follows:

$$\mathbf{h} = \sum_{i \in \mathcal{V}} \operatorname{softmax}\left(\mathcal{F}_{\text{gate}}(h_i)\right) \odot \left(\operatorname{LayerNorm}(\mathcal{F}_{\mathbf{\Theta}}(h_i)) + h_i\right) \tag{4.5}$$

where $\mathcal{F}_{\text{gate}}$ and $\mathcal{F}_{\mathbf{\Theta}}$ are neural networks, and $softmax\left(\mathcal{F}_{\text{gate}}(h_n)\right)$ serves as a soft attention mechanism that determines the nodes relevant to the current task. Specifically, we use $\mathcal{F}_{\text{gate}}$ to compute attention scores and $\mathcal{F}_{\mathbf{\Theta}}$ to map node features and then combine them with the attention scores. Additionally, we added a skip connection to prevent our model from over-smoothing. Moreover, the soft-attention aggregator has the added benefit of effectively filtering out unimportant information that may be present. We have observed experimentally that such an aggregation method gives better results than a simple sum or mean.

Our main insight is that, while the attention scores from the Equation 4.1 capture broader, long-range dependencies across distant regions of the graph, the residual soft-attention and the gating mechanism introduced in Equation 4.2 contribute to the modeling of more localized information, thus enhancing learned graph representation.

Figura 11 – The complete architecture, including the proposed message passing operator. The initial layer receives and processes the node feature matrix. Next, the message-passing operators work with the node feature and adjacency matrices. After that, we implement layer normalization to ensure that the activations are standardized across the features in each layer. The graph pooling layer is then employed to summarize the node embeddings at the graph level. Finally, the final multilayer perceptron (MLP) in the architecture generates a logit that reflects the probability of a link existing between two specific nodes.



Source: Thesis author

The complete architecture, which incorporates the proposed Gated Graph Transformer operator, is illustrated in Figure 11. The first layer takes in and processes the node feature

matrix. The message-passing operators then handle the embedded node feature and adjacency matrices. We then apply layer normalization to standardize the activations across the features within each layer. The graph pooling layer creates a graph-level summary of the node embeddings. Finally, the last multi-layer perceptron (MLP) in the architecture produces a logit that indicates the likelihood of a link existing between two specific nodes.

4.3.4 DETERMINING THE THRESHOLD

We apply a threshold to determine what is an anomaly and what is benign. Regarding the threshold choice, we proceed as in (KING; HUANG, 2023). Specifically, given a set of scores given for edges that exist and a set of scores for non-edges in the validation data, the optimal cutoff threshold τ is the one that achieves optimal TPR and FPR and satisfies

$$\arg\min_{\tau} \parallel (1-\lambda)\text{TPR}(\tau) - \lambda\text{FPR}(\tau) \parallel \tag{4.6}$$

where τ is a cutoff threshold, and λ is a hyperparameter in $[0-1]$, biasing the model to optimize for either a high true positive rate or a low false positive rate.

Moreover, $\mathrm{TPR}(\tau)$ and $\mathrm{FPR}(\tau)$ refer to the true and false positive rate of classification given cutoff threshold τ. We use $\lambda = 0.5$ in all experiments as such value has shown to be more effective. Crucially, our aim with such an approach is to automate the process of deciding the threshold through an optimization process whose inputs are the scores for existing edges and non-edges. Since it uses both positive and negative edge sets as input, this approach is generic and effectively applicable to other scenarios that adopt the negative sampling strategy for link prediction for cybersecurity purposes.

4.4 EXPERIMENTAL SETUP FOR GATED-GRAPH TRANSFORMER

The models were trained using focal loss and Adam optimizer with a learning rate of 0.0001. Moreover, we use 10% of edges for testing and 5% for validation. The remaining is reserved for training.

4.4.1 DATASETS

This study evaluates the proposed method using the CERT Insider Threat Dataset (GLASSER; LINDAUER, 2013) and PicoDomain (LAPRADE; BOWMAN; HUANG, 2020). The CERT dataset is a publicly available dataset that can be used to research, develop, and test methods for mitigating insider threats.

The CERT Insider Threat Center defines insider threats as threats that can be carried out by malicious or unintentional insiders. These insiders have authorized access to the organization's network, systems, and data. They can exploit this access to negatively affect the confidentiality, integrity, and availability of the organization's information. Intentional insider threats are carried out by malicious insiders, who may sabotage information systems, steal intellectual property, or disclose classified information. Unintentional insider threats are carried out by users who are negligent in using authorized resources, such as by clicking on phishing links or downloading malicious files.

The dataset consists of multiple releases, each characterizing an organization with 1,000 to 4,000 employees. The CERT 4.2 contains activity logs from 1000 users and 1003 computers. Release 5.2 of the dataset (CERT r5.2) simulates an organization with 2000 employees over 18 months. Each malicious insider in CERT r5.2 belongs to one of four popular insider threat scenarios: data exfiltration (scenario 1), intellectual property theft (scenarios 2, 4), and IT sabotage (scenario 3).

The CERT dataset contains multisource activity logs such as user login/logoff, emails, file access, website visiting, removable device usage, and organizational structure. These events are present in separate CSV files. In this work, we use only user login activities (in the file *logon.csv*) for both CERT r4.2 and r5.2.

In turn, the PicoDomain consists of 3 days of detailed Zeek logs collected in a simulated small-scale network where APT attacks occurred in the last two days. Zeek is a passive, open-source network traffic analyzer that is widely used by operators to examine network traffic packets. Zeek logs are detailed, structured records generated by the tool, which can help detect malicious activities within a network. These logs capture a wide range of network activities, providing a comprehensive overview of network behavior.

Tabela 1 – We compared the performance of all approaches in our experiments. In comparing ten different models, we discovered that the proposed model excels above the majority. It demonstrates exceptional performance in various metrics across the datasets we utilized. Furthermore, the proposed model has a lower rate of False Positives, showing even greater significance in our study.

Dataset	Model	AUC	TPR	FPR	TP	FP	F1
CERT r4.2	Gated-GraphTransformer	94.3859 ± 0.0817	92.5330 ± 0.2136	**7.7420 ± 0.2878**	224300.0000 ± 517.6871	18766.6660 ± 697.6150	92.4356 ± 0.0878
	DGCNN	94.3274 ± 0.0718	92.9730 ± 0.3255	9.6947 ± 0.8434	225366.6719 ± 789.0923	23500.0000 ± 2044.5049	92.1470 ± 0.1383
	GAT	94.7891 ± 0.0860	92.8287 ± 0.6895	12.6100 ± 4.7206	225016.6719 ± 1671.4265	30566.6660 ± 11442.8438	92.2165 ± 0.1676
	GATV2	94.9026 ± 0.1193	92.9112 ± 0.4960	11.6887 ± 2.2485	225216.6719 ± 1202.3588	28333.3340 ± 5450.3823	92.3303 ± 0.1280
	GCN	94.3243 ± 0.0770	93.5369 ± 0.3338	14.8790 ± 2.6842	226733.3281 ± 809.1147	36066.6680 ± 6506.5098	92.1692 ± 0.1219
	GIN	94.2180 ± 0.1505	92.5055 ± 0.3171	8.4158 ± 0.5059	224233.3281 ± 768.5484	20400.0000 ± 1226.3768	92.1817 ± 0.1093
	GatedGraphConv	94.5331 ± 0.1744	92.7462 ± 0.1816	8.2508 ± 0.6130	224816.6719 ± 440.0757	20000.0000 ± 1485.9341	92.3995 ± 0.1062
	Node2Vec	49.1371 ± 0.0421	49.6837 ± 0.4716	52.0558 ± 0.4210	120433.3359 ± 1143.0952	126183.3359 ± 1020.6207	48.9540 ± 0.0682
	SAGE-add	94.4155 ± 0.2872	93.2068 ± 0.5496	11.1799 ± 3.4720	225933.3281 ± 1332.1661	27100.0000 ± 8416.1748	92.2948 ± 0.0683
	SAGE-max	94.4824 ± 0.1415	92.5055 ± 0.3536	8.2646 ± 0.3478	224233.3281 ± 857.1270	20033.3340 ± 843.0105	92.3500 ± 0.0724
	SAGE-mean	94.8133 ± 0.2267	93.1724 ± 0.1447	9.4266 ± 0.3287	225850.0000 ± 350.7136	22850.0000 ± 796.8689	92.5371 ± 0.0878
CERT r5.2	Gated-GraphTransformer	95.3451 ± 0.1306	93.5209 ± 0.2358	**6.8121 ± 0.3935**	613216.6875 ± 1545.8546	44666.6680 ± 2580.4392	93.5225 ± 0.0564
	DGCNN	95.3275 ± 0.1365	93.7243 ± 0.1633	8.2024 ± 0.8344	614550.0000 ± 1070.9808	53783.3320 ± 5471.1670	93.3895 ± 0.0451
	GAT	95.6779 ± 0.0764	93.1396 ± 0.3934	9.1480 ± 1.3522	610716.6875 ± 2579.4702	59983.3320 ± 8866.6602	93.3886 ± 0.1589
	GATV2	95.5954 ± 0.1269	93.6200 ± 0.3034	9.2319 ± 2.0476	613866.6875 ± 1989.6399	60533.3320 ± 13426.1934	93.5845 ± 0.0852
	GCN	95.0554 ± 0.1411	93.8844 ± 0.2051	12.2947 ± 1.7581	615600.0000 ± 1344.6189	80616.6641 ± 11527.9512	93.3530 ± 0.0636
	GIN	95.3148 ± 0.1684	93.5133 ± 0.2077	8.0880 ± 0.6857	613166.6875 ± 1361.8615	53033.3320 ± 4496.0723	93.4028 ± 0.0741
	GatedGraphConv	95.5418 ± 0.0517	93.3532 ± 0.1978	**6.6011 ± 0.4153**	612116.6875 ± 1296.7909	43283.3320 ± 2722.8049	93.5664 ± 0.0659
	Node2Vec	50.1118 ± 0.1930	50.2389 ± 0.0685	49.8424 ± 0.4172	329416.6562 ± 449.0731	326816.6562 ± 2735.2634	49.9912 ± 0.0248
	SAGE-add	95.5118 ± 0.0828	93.8920 ± 0.1919	7.9788 ± 0.8703	615650.0000 ± 1258.1733	52316.6680 ± 5706.6338	93.5873 ± 0.0572
	SAGE-max	95.4891 ± 0.1164	93.5794 ± 0.1585	6.8705 ± 0.4554	613600.0000 ± 1039.2305	45050.0000 ± 2985.7998	93.5708 ± 0.0501
	SAGE-mean	95.6896 ± 0.1280	93.8336 ± 0.0615	7.2416 ± 0.1624	615266.6875 ± 403.3195	47483.3320 ± 1064.7378	93.6942 ± 0.0290
Pico Domain	Gated-GraphTransformer	75.0000 ± 0.0000	100.0000 ± 0.0000	**25.0000 ± 0.0000**	400.0000 ± 0.0000	100.0000 ± 0.0000	88.8889 ± 0.0000
	DGCNN	60.4167 ± 28.6865	70.8333 ± 29.2261	79.1667 ± 18.8193	283.3333 ± 116.9045	316.6667 ± 75.2773	59.8545 ± 27.6942
	GAT	53.1250 ± 25.8451	66.6667 ± 43.7798	70.8333 ± 29.2261	266.6667 ± 175.1190	283.3333 ± 116.9045	52.1212 ± 30.1499
	GATV2	57.2917 ± 30.9780	91.6667 ± 12.9099	41.6667 ± 20.4124	366.6667 ± 51.6398	166.6667 ± 81.6497	74.4529 ± 13.2439
	GCN	56.2500 ± 11.1803	79.1667 ± 24.5798	62.5000 ± 34.4601	316.6667 ± 98.3192	250.0000 ± 137.8405	51.2963 ± 28.2005
	GIN	57.2917 ± 17.4180	83.3333 ± 20.4124	66.6667 ± 30.2765	333.3333 ± 81.6497	266.6667 ± 121.1060	57.2222 ± 13.4026
	GatedGraphConv	45.8333 ± 27.8575	70.8333 ± 40.0520	50.0000 ± 38.7298	283.3333 ± 160.2082	200.0000 ± 154.9193	53.6508 ± 35.2773
	Node2Vec	70.8333 ± 10.2062	70.8333 ± 10.2062	29.1667 ± 10.2062	283.3333 ± 40.8248	116.6667 ± 40.8248	70.8333 ± 10.2062
	SAGE-add	42.7083 ± 19.1281	95.8333 ± 10.2062	62.5000 ± 34.4601	383.3333 ± 40.8248	250.0000 ± 137.8405	69.8990 ± 5.5104
	SAGE-max	40.6250 ± 21.5602	95.8333 ± 10.2062	75.0000 ± 31.6228	383.3333 ± 40.8248	300.0000 ± 126.4911	65.7576 ± 8.5345
	SAGE-mean	55.2083 ± 10.0130	95.8333 ± 10.2062	75.0000 ± 31.6228	383.3333 ± 40.8248	300.0000 ± 126.4911	60.1010 ± 14.3953

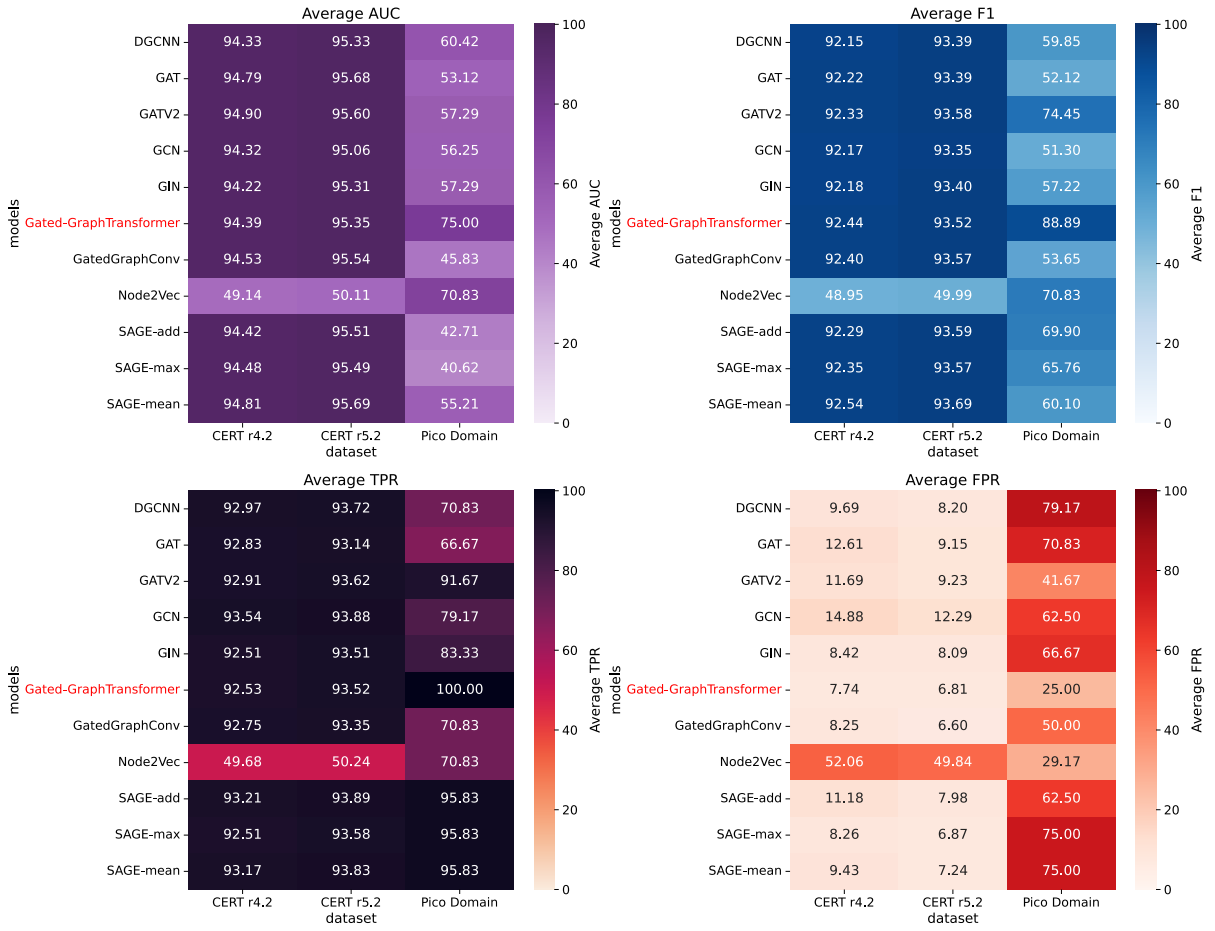
4.4.2 EVALUATION METRICS

To evaluate our technique for detecting malicious authentication in enterprise networks, we report the true positive rate (TPR) and false positive rate (FPR), which is consonant with practices in the literature (BOWMAN et al., 2020; KING; HUANG, 2023). Additionally, we include AUC and F1 scores. The experiments are repeated five times, and we report the average results.

4.5 RESULTS AND DISCUSSIONS

Our experiments involved comparing ten models on three complementary datasets. The results showed that our model performed better than most of the others regarding false positives. Table 1 displays the results, and it can be seen that our model achieved the best FPR score, outperforming state-of-the-art approaches such as GAT (CASANOVA; LIO; BENGIO, 2018), GCN (KIPF; WELLING, 2017), and SAGE (HAMILTON; YING; LESKOVEC, 2017). Additionally, we

Figura 12 – Comparative performance of the evaluated models on multiple metrics. While the overall results are comparable for most metrics, our proposed model demonstrates a superior ability to minimize false positives, surpassing other examined approaches. Notably, in the Pico Domain dataset, a high F1 score proved elusive for all models, suggesting a higher complexity and challenge inherent in this dataset. Despite this, our Gated Graph Transformer model achieved the most meritorious performance on this particularly demanding dataset, outperforming the evaluated models.



compare the performances in Figure 12 for better visualization.

The FPR metric, which measures the proportion of benign samples that are incorrectly classified as malicious, is an essential concern in cybersecurity and holds substantial significance as a Key Performance Indicator (KPI) within the realm of threat intelligence. Fundamentally, this particular metric serves to quantify the number of false alarms generated by tools and processes utilized in the field of threat intelligence. A heightened FPR may indicate the ineffectiveness of said tools, while a diminished FPR is generally deemed acceptable within the security industry.

As we can note from the results summarized in Table 1 and Figure 12, our model's performance in FPR is consistently better, outperforming most of the compared approaches. Although our method performed poorly on PicoDomain compared to its performance on CERT,

primarily concerning the false positive rate (FPR) metric, its results were still superior to those of competing methods on the same dataset.

4.5.1 CONTRIBUTION OF DIFFERENT AGGREGATION METHODS

Tabela 2 – The results show the average performance of the models (in percentage) using the proposed aggregation function and traditional methods over five runs. We want to highlight the performance of models in the FPR metric, which is a crucial metric in cybersecurity.

dataset	aggregation	AUC	TPR	FPR
CERT R4.2	proposal	94.3859 ± 0.0817	92.5330 ± 0.2136	**7.7420 ± 0.2878**
	add	95.0016 ± 0.2227	93.2550 ± 0.2721	8.5877 ± 0.4778
	max	95.1078 ± 0.0993	93.0624 ± 0.3943	8.0652 ± 0.3084
	mean	94.6605 ± 0.1420	92.9387 ± 0.3524	8.0789 ± 0.2910
	power	94.6605 ± 0.1420	92.9387 ± 0.3524	8.0789 ± 0.2910
CERT R5.2	proposal	95.3451 ± 0.1306	93.5209 ± 0.2358	**6.8121 ± 0.3935**
	add	95.5101 ± 0.0289	93.7802 ± 0.0792	6.8400 ± 0.2328
	max	95.4969 ± 0.0978	93.8285 ± 0.1304	7.7373 ± 1.0356
	mean	95.3872 ± 0.0417	93.8158 ± 0.0904	7.2823 ± 0.4479
	power	95.3872 ± 0.0417	93.8158 ± 0.0904	7.2823 ± 0.4479
PicoDomain	proposal	75.0000 ± 0.0000	100.0000 ± 0.0000	**25.0000 ± 0.0000**
	add	76.0417 ± 2.5516	100.0000 ± 0.0000	41.6667 ± 30.2765
	max	75.0000 ± 0.0000	100.0000 ± 0.0000	37.5000 ± 30.6186
	mean	75.0000 ± 0.0000	100.0000 ± 0.0000	45.8333 ± 33.2290
	power	75.0000 ± 0.0000	100.0000 ± 0.0000	45.8333 ± 33.2290

To examine the impact of the aggregation method proposed in this paper, we evaluated the performance of our model using various aggregation methods. As shown in Table 2, regarding the false positive rate, the residual soft-attention aggregation method has been shown to outperform competing approaches in all three datasets used.

Concerning the AUC and TPR metrics, the performance of all aggregation methods is predominantly similar. This may suggest that the sensitivity of the models is invariant with respect to the aggregation technique employed. Alternatively, it may indicate that the dataset has specific characteristics that result in the uniform performance of all models in terms of those metrics. Nevertheless, in practice, we argue that the model that presents the lowest false positive rates while maintaining high predictive performance in the other metrics is generally the preferred solution. As previously discussed, a low FPR is desirable, indicating that the method is less likely to generate false alarms.

The residual soft-attention aggregation method can produce more powerful models that are less likely to be affected by aggregating less informative structural representations of nodes. On

the other hand, the sum and mean aggregation methods, for example, simply combine the node representations without considering their relative importance. This can lead to aggregating less informative structural representations of nodes, negatively impacting the model's performance.

Figura 13 – A comparison of FPR performances for various node aggregation methods concerning the depth of the GraphTransformer model on the "r4.2"and "r5.2"datasets. The x-axis indicates the depth of each model through the number of layers; the y-axis indicates the false positive rate. The results show that the proposed node aggregation method consistently outperforms others, even in deeper models. This outcome suggests that our model is less affected by over-smoothing than models based on more traditional aggregation methods.

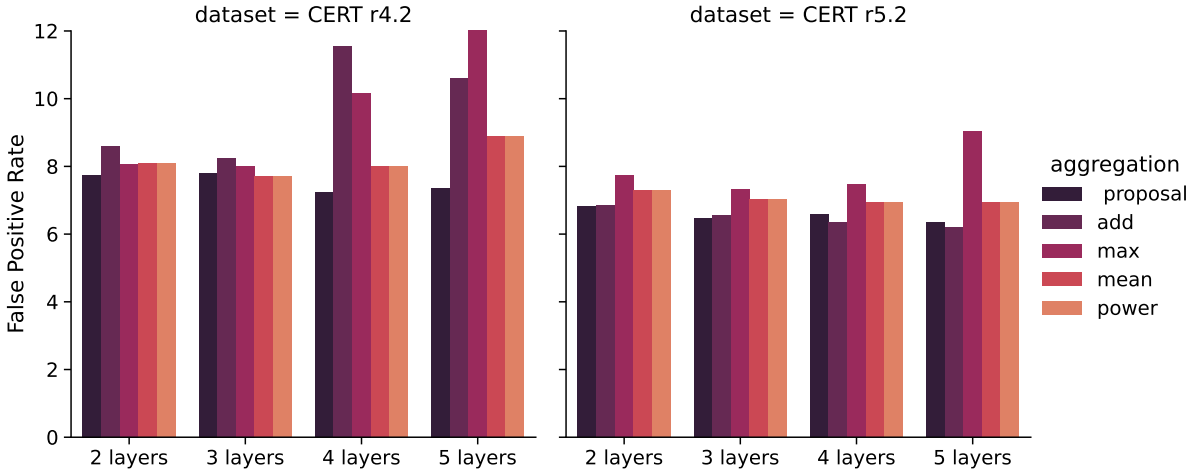


Figure 13 depicts a comparison of FPR performances among different node aggregation methods concerning the depth of the GraphTransformer model on the "r4.2"and "r5.2"datasets. Evidently, the proposed aggregation method consistently outperforms others, even in deeper models. This outcome suggests that our model is less affected by over-smoothing than models based on more traditional aggregation methods.

Moreover, we assessed the complexity of the proposed aggregation method in terms of the time required to train each GraphTransformer model on the "r4.2"and "r5.2"datasets, as illustrated in Figure 14. Notably, the training time for our model generally exceeds that of models employing more straightforward aggregation methods. This discrepancy was expected since aggregation using simpler functions like maximum, average, or sum values does not significantly impact the model's parameter count, unlike the proposed approach that involves computing more parameters.

This trade-off between training time and performance is a common consideration in model development, but the advantages of our model outweigh the increased training time. Nevertheless, we plan to explore optimization techniques to mitigate our architecture's complexity for future research.

Figura 14 – The complexity of the proposed aggregation method in terms of the time (average in seconds from five executions) required to train each two-layer GraphTransformer model on the “r4.2"and “r5.2"datasets compared to more traditional aggregation methods. The x-axis indicates the method used for node aggregation; the y-axis indicates the average time in seconds used to train each model. Our approach delivers significantly improved performance despite requiring longer training than simpler models. We believe further optimization efforts can bridge this gap while maintaining these gains.

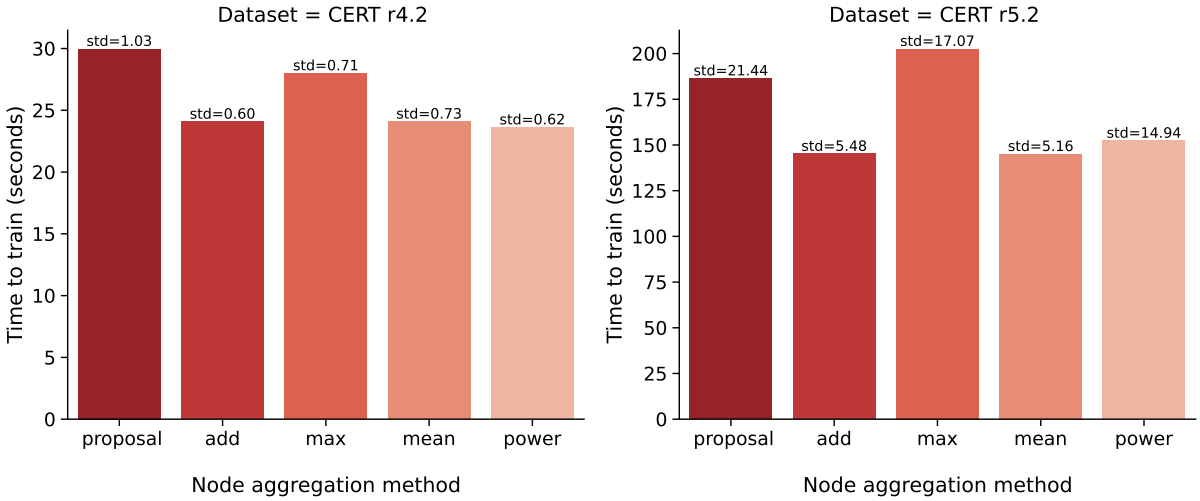
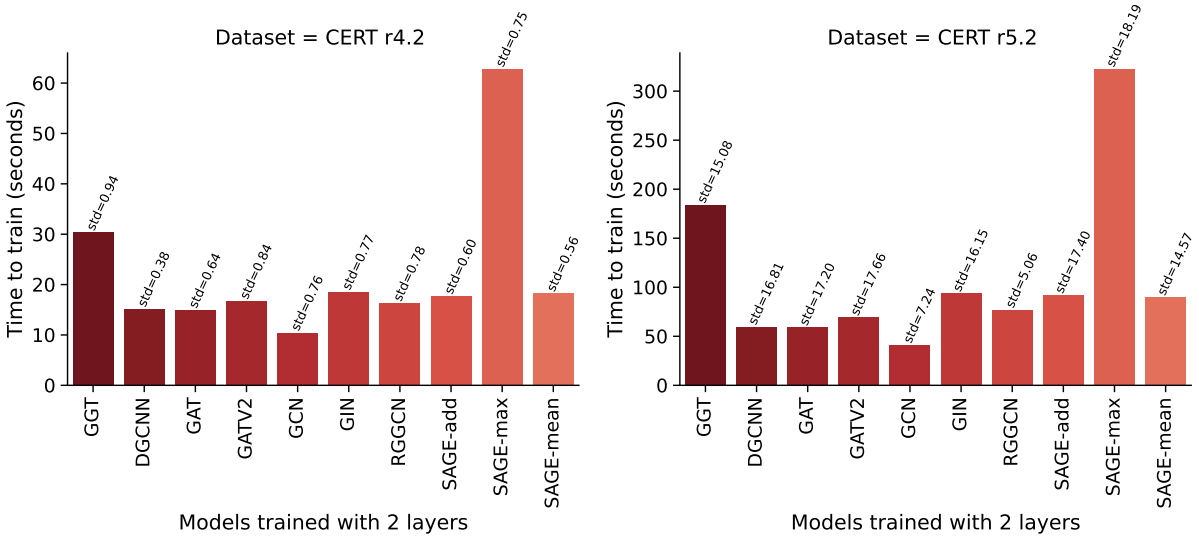


Figura 15 – A comparison between the training time (average in seconds from five executions) required for our model and that of other GNN architectures. The x-axis represents the different architectures; the y-axis represents each model’s average training time in seconds. For better visualization, GGT is short for our Gated-GraphTransformer. Despite our model requiring a longer training time than other GNN architectures, it’s important to remember that our proposal also delivers significantly improved performance with lower false positive rates.

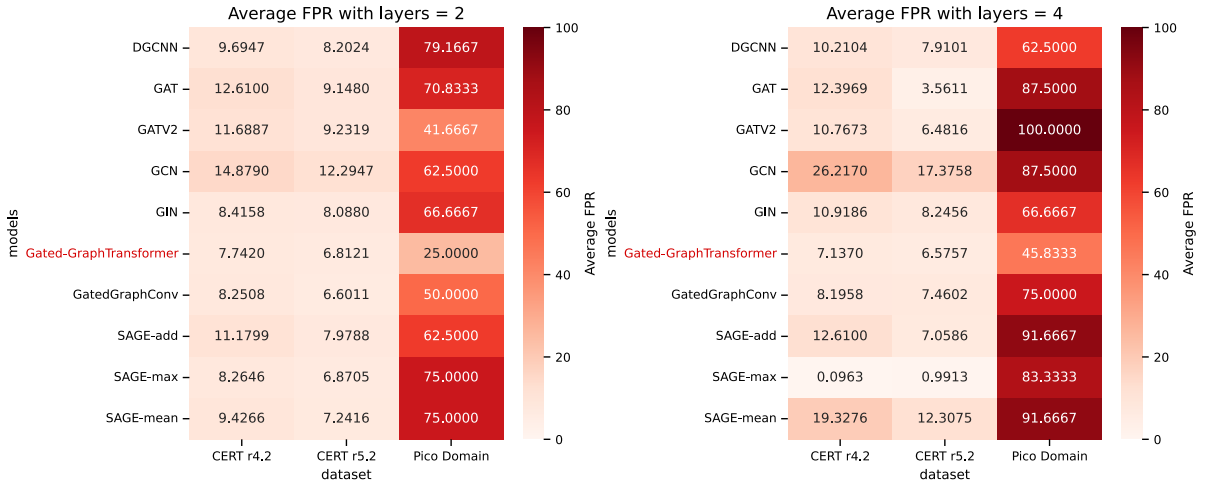


Finally, Figure 15 compares the training time required for our model and that of other GNN architectures. In part, our model requires more training time than other architectures due to inheriting the computational complexity inherent in the vanilla Transformer architecture

(KITAEV; KAISER; LEVSKAYA, 2019; LIN et al., 2022), upon which we have based our approach.

4.5.2 WILL ADDING MORE LAYERS AFFECT THE FPR PERFORMANCE?

Figura 16 – The false positive rate in various models tends to be sensitive to an increase in the number of layers. Notably, DGCNN, GCN, GIN, SAGE-add, SAGE-mean, and GatedGraphConv(BRESSON; LAURENT, 2018) experience a significant increase in the false positive rate (FPR), while GAT and SAGE-max demonstrate noteworthy improvements, though less consistently. In the “r4.2"and “r5.2"datasets, our model surpasses most competing methods, even with four layers, where over-smoothing is expected.



Regarding the false positive rate, models commonly exhibit sensitivity to an increase in the number of layers, as depicted in Figure 16. Some models, such as DGCNN, GCN, GIN, SAGE-add, SAGE-mean, and GatedGraphConv(BRESSON; LAURENT, 2018), experience a significant rise in the false positive rate (FPR), whereas others, like GAT and SAGE-max, show notable improvements, albeit less frequently. Across the PicoDomain dataset, all models experience a substantial deterioration in the FPR. Nevertheless, the proposed model consistently outperforms others in performance. In the CERT “r4.2"and “r5.2"datasets, our model surpasses most competing methods, even when using four layers, where over-smoothing is expected to occur.

4.5.3 HOW DOES THE PROPOSED GNN COMPARE TO EXISTING STATE-OF-THE-ART APPROACHES

Not many works explore the problem of detecting anomalous authentications to discover lateral movement through Graph Neural Networks in the context of link prediction. Nonetheless, our literature review found some significant works in the context (DUAN et al., 2020; SUN;

YANG, 2022; KING; HUANG, 2023). For example, the study in (KING; HUANG, 2023) presents a model-agnostic framework for detecting lateral movements using GNNs in temporal graphs. The aim is to capture temporal relations of data to perform anomaly link prediction. In (DUAN et al., 2020), the authors propose an anomaly aware network embedding (AANE) model based on a Graph Auto-Encoders (GAE) with a tailored loss function comprising an anomaly aware loss and adjusted fitting loss and use it as an anomaly indicator to select significant anomalous links during model training iteratively.

The work in (SUN; YANG, 2022) introduces HetGLM, a GNN-based anomaly link prediction algorithm to detect lateral movements. HetGLM utilizes a graph auto-encoder with a metapath-based neighbor sampling strategy and attention mechanism. For the decoder part, a dual-decoder structure, working together with a mutual information (MI) regularization term, is designed to identify anomaly links via relative reconstruction errors. The authors have presented different versions of the HetGLM. Specifically, *HetGLMoA* describes HetGLM without the attention mechanism; *HetGLMoD* adapts HetGLM to use a traditional decoder instead of a dual-decoder structure. Finally, *HetGLMoR* defines HetGLM without the mutual information regularization term. The method GLGV introduced by (BOWMAN et al., 2020) generates node embeddings using DeepWalk. It represents links as Hadamard products and detects lateral movement with a logistic regression classifier.

Tabela 3 – Comparison of the proposed approach with existing state-of-the-art models. We compare our results with those extracted from (SUN; YANG, 2022). The comparison is based on the F1 metric.

Model	**CERT r4.2**	**PicoDomain**
HetGLMoA (SUN; YANG, 2022)	89.68	92.18
HetGLMoD (SUN; YANG, 2022)	83.84	89.85
HetGLMoR (SUN; YANG, 2022)	89.99	91.72
HetGLM (SUN; YANG, 2022)	91.28	92.68
GLGV (BOWMAN et al., 2020)	82.75	81.86
AANE (DUAN et al., 2020)	88.58	90.66
Gated-Graph Transformer	**92.43**	88.88

We compare our results with HetGLM, AANE, and GLGV, as their authors partly used the same datasets as our study. The comparison is based on the F1 metric, the only common metric reported in their paper. As shown in Table 3, the Gated-Graph Transformer achieves a higher F1 on the CERT r4.2 dataset, indicating that it is effective for anomaly detection. In the PicoDomain dataset, the proposed model is very competitive, maintaining significant robustness even in different data contexts, which the other models do not present. However, the

lower F1 score of the Gated-Graph Transformer on the PicoDomain dataset may be attributed to its significantly compact nature when compared to CERT R4.2. This compactness could result in less useful information being available to the Gated-Graph Transformer than CERT R4.2.

4.5.4 KEY ASPECTS OF THE PROPOSED APPROACH.

Finally, we would like to highlight the following points regarding the results of our experiments:

Model Robustness. Regarding the model robustness, the *Gated-GraphTransformer* consistently outperforms other models across all datasets, suggesting it might be a robust model for tasks of this nature, regardless of dataset variations.

Dataset Complexity. The Pico Domain dataset appears more challenging for most models than the CERT dataset. This could point to inherent complexities or nuances in the Pico Domain data that might warrant further investigation.

Trade-offs. While our model achieves superior performance, it comes at the cost of longer training times compared to simpler aggregation methods. This is unsurprising as our more complex architecture necessitates computationally demanding functions. Moreover, while the ability of graph transformers to model all pairwise interactions through a full attention mechanism is their hallmark, it also leads to quadratic computational and memory barriers. Sparse attention frameworks, specifically designed for graph data, hold significant promise towards optimizing these models, which have been a critical research priority, as stated by authors in (SHIRZAD et al., 2023). We intend to investigate optimization techniques to reduce this training overhead.

Model Variability. High standard deviations in model performance metrics for specific models, like GAT, SAGE, and GIN, suggest that these models might be sensitive to variations in data. This could point to potential overfitting or instability in the model's learning process.

Potential for Improvement. The results reveal areas of potential improvement, especially for models with high False Positive Rates on specific datasets. This could open avenues for research into model refinement or hyperparameter tuning.

In this work, we study the problem of identifying abnormal authentications in computer networks. Atypical logons can indicate unauthorized access, which generally reveals an ongoing attack. However, the problem imposes unique challenges, as attackers can use legitimate

credentials to access the machines. Therefore, solutions that adapt to attackers' changing behavior are required.

To identify atypical logons, we model the problem as discovering low-probability links in a graph. In this graph, users and computers are represented by nodes, and authentication events are denoted by edges. We then use GNNs to learn the graph structure and identify those links.

To uncover unauthorized access to machines within a computer network, we focused on proposing a new GNN architecture to predict anomalous logins, which are built upon the multi-head attention mechanism derived from transformers. Essentially, the proposal incorporates two crucial mechanisms. Firstly, we introduce a residual soft-attention scheme that facilitates the aggregation of node representations through a weighted sum, resulting in enhanced node representations and improved filtration of irrelevant information. Secondly, we incorporate a gating mechanism that aids the model in determining the relative importance of edges during the message-passing process.

Our proposed method outperforms the state-of-the-art in anomalous link detection with significantly improved FPR at the cost of longer training times than other models. This trade-off between training time and performance is a common consideration in model development. Reducing false positives justifies the training time difference, especially in cybersecurity applications, where lower FPR is paramount. We believe further optimization efforts can bridge this gap while maintaining these gains. Sparse attention frameworks, specifically designed for graph data, hold significant promise in this direction.

5 SECOND APPROACH – DETECTING SUSPICIOUS ENDPOINTS IN COMPUTER NETWORKS VIA SCALABLE SPATIOTEMPORAL GRAPH TRANSFORMERS FOR NODE CLASSIFICATION

When managing large and complex networks, additional layers of protection are essential. While lateral movement detection methods, such as the approach proposed in Chapter 4, can help hinder an adversary's ability to establish a presence across the network, detecting and isolating suspicious endpoints—including those involved in other APT phases—can further enhance security by providing additional protection layers. Understanding the node roles within the network is a key step for that purpose.

GTs are well-suited for node classification in large-scale networks due to their ability to capture long-range dependencies and aggregate global information. Unlike traditional GNNs, GTs enable each node to attend to all other nodes, allowing for information aggregation from distant nodes and overcoming GNN limitations in capturing long-range dependencies, particularly in the context of large graphs.

However, scaling algorithms for node classification in large graphs, like those commonly encountered in cybersecurity, pose two significant challenges. First, the quadratic complexity of self-attention in standard GTs makes them computationally infeasible for large datasets. Second, allowing each node to attend to all others results in noise aggregation from irrelevant nodes and leads to ambiguous attention weights. Furthermore, existing techniques neglecting time-based data may miss significant details, as anomalies are often part of a sequence of events rather than isolated incidents, requiring considering both spatial and temporal contexts for accurate identification.

This chapter explores the application of efficient GTs for node classification in the context of anomaly detection to detect anomalous endpoints (hosts) involved in abnormal activities. Essentially, we use node classification to understand the role of each node within the network, allowing us to detect and isolate suspicious hosts.

In particular, we focus on recent advancements in attention mechanisms that aim to overcome the scalability challenges associated with traditional graph transformers. To further improve the model quality by effectively reducing the influence of irrelevant data that often hamper the accuracy of graph transformer-based node classification, our approach employs a soft-attention mechanism applied to node representations, enabling the model to selectively prioritize relevant connections within the graph. Our approach incorporates temporal infor-

mation into different granularity levels. Specifically, we leverage short-temporal dependencies through node features and contextual time embeddings by including edge features.

Experimental results demonstrate our approach consistently outperforms existing GNNs in terms of both generalization and stability. While achieving ROC-AUC results superior to a significant portion of competitors, it remains at least competitive when not surpassing them.

5.1 THE NEED FOR TEMPORAL DYNAMICS IN ANOMALY DETECTION

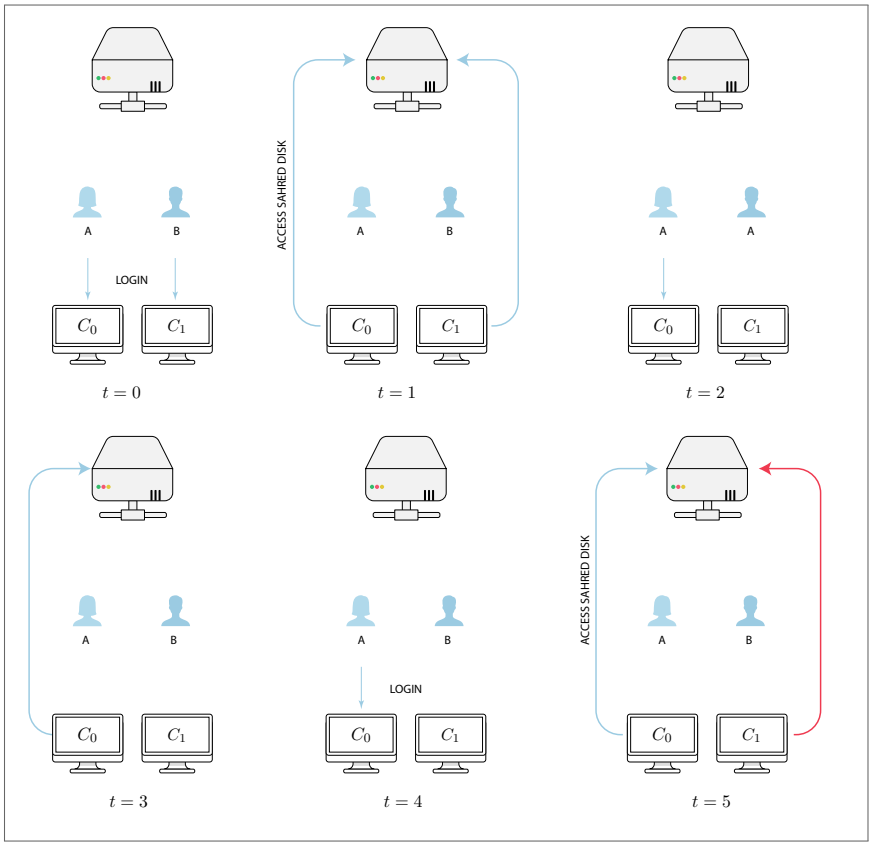
While adept at leveraging graph structure, most traditional GNN architectures struggle to incorporate edge features effectively into their node representations. In many real-world graphs, the edges hold crucial information about node interactions. GNNs can derive more nuanced and informative node representations by considering edge features. This is because the edges provide context about how a node relates to others in the network, which can be critical for tasks like anomaly detection.

Moreover, as pointed out by authors in (WU et al., 2022) and (CHEN; TAO; WONG, 2021), existing GNNs face other limitations, particularly in the context of large graphs, where issues like over-smoothing, over-squashing, and the inability to capture long-range dependencies hinder their performance. These limitations are especially pronounced in cybersecurity applications, where the ability to capture global context and detect subtle, long-range interactions is critical for identifying anomalous behavior, such as Advanced Persistent threats (APTs) (ALSHAMRANI et al., 2019), a class of sophisticated attacks launched by resourceful adversaries using a wide spectrum of attack techniques and tools.

Furthermore, existing techniques that typically neglect the temporal dynamics of data (ZOLA et al., 2022; GONçALVES; ZANCHETTIN, 2024a; SUN; YANG, 2022; POWELL, 2020), and analyzing graph structures without a time-based perspective may only uncover general trends that have limited significance for cybersecurity threats. In fact, anomalies are rarely isolated events. To accurately identify them, we must consider the spatial relationships and the temporal context of an entity's interactions (KING; HUANG, 2023).

To illustrate the need for considering temporal context, consider the example in Figure 17. This figure depicts a network with two users, *A* and *B*, and three devices, C_0, C_1, and a shared drive. The first two time slices show typical network behavior. At time t_0, users *A* and *B* authenticate with their respective computers, C_0 and C_1. Subsequently, at $t = 1$, both computers access the shared drive. During time slices $t = 2$ and $t = 3$, we observe that

Figura 17 – This figure illustrates a challenging scenario for detecting anomalous activity in a network. It depicts a network with two users (A and B) and three devices (C_0, C_1, and a shared drive). The typical sequence of events is for a user to authenticate with a computer, followed by that computer accessing the shared drive. However, at time $t = 5$, computer C_1 accesses the shared drive without user B having first authenticated. This suggests a potential malicious process running on C_1, operating outside the control of the machine's primary user.



Source: Thesis author. Adapted from (KING; HUANG, 2023).

when user B does not first authenticate with computer C_1, the latter does not communicate with the shared drive. This demonstrates a normal pattern where access to the shared drive is contingent on user authentication.

However, at times $t = 4$ and $t = 5$, an atypical pattern is observed: computer C_1 accesses the shared drive without prior authentication by user B. This deviation from expected behavior may indicate potential malicious activity, such as remote service hijacking (T1563) or an attempt to compromise shared content (T1080), as categorized in the MITRE ATT&CK framework (ATT&CK®, 2018). Both techniques are commonly employed during APT attacks.

Detecting such attacks requires a model that understands the temporal context of events. Simply looking at an individual event in isolation is insufficient. The model must consider the sequence of events that led to it and the broader network interactions occurring simultaneously. For example, a connection between a computer and a shared drive might be considered normal in one scenario but suspicious in another if the user hasn't authenticated.

Current graph-based approaches, which lack a time dimension, and many event-based approaches (ALMEIDA et al., 2023; GONçALVES; ZANCHETTIN, 2024b; BIAN et al., 2021) that treat each event independently, wouldn't be able to discern the difference between the computer accessing the shared drive at time $t = 1$ (following user authentication) and at time $t = 5$ (without authentication).

We propose representing the network as a temporal continuous-time graph, where each edge is associated with a timestamp indicating when it was formed. This approach allows for a detailed modeling of interactions, capturing the network's structure at every moment while preserving the dynamic nature of the connections over time. By doing so, we can analyze how these connections evolve and identify anomalies based on their temporal context.

5.2 CHALLENGES OF SCALING GRAPH TRANSFORMERS

The global attention mechanism in Graph Transformers enables them to capture relationships between any two nodes, regardless of their distance within the graph. This global perspective makes GTs well-suited for tasks requiring a comprehensive understanding of network interactions, such as anomaly detection in large-scale networks. In particular, the attention module at the heart of every Transformer architecture is responsible for computing pairwise similarity scores between each position in an input sequence. However, this approach has a major scalability limitation: it requires a significant amount of computational resources and memory to compute all these similarity scores in parallel, leading to quadratic complexity with respect to the length of the input sequence (TAY et al., 2022). Specifically, this results in both computationally expensive processing time and substantial memory requirements to store the resulting matrix of similarity scores. That quadratic complexity poses significant challenges when scaling to large graphs.

5.3 OUR CONTRIBUTION: EXPLORING EFFICIENT GRAPH TRANSFORMERS FOR DETECTING UNAUTHORIZED ACCESS TO COMPUTER NETWORKS

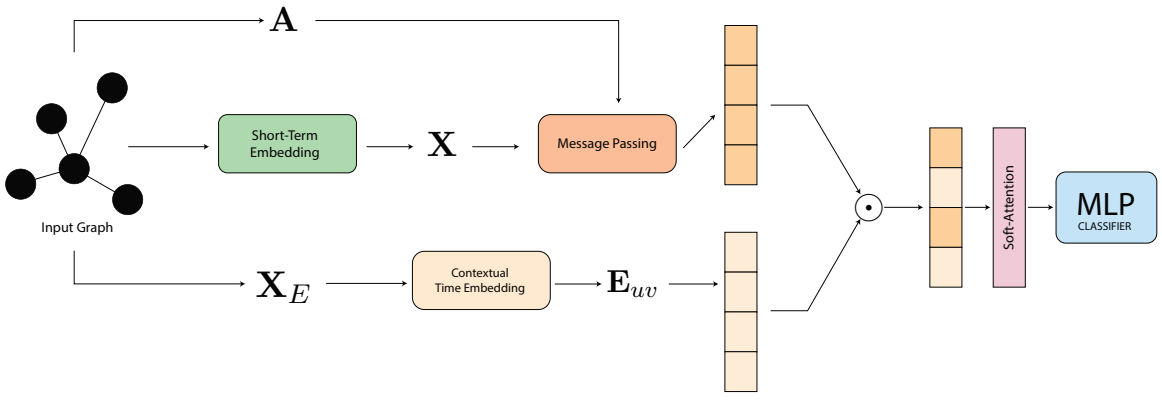
Graph Transformers are particularly well-suited for node classification in large-scale networks due to their ability to capture long-range dependencies and aggregate global information. Detecting the node responsible for malicious activity is critical to prevent further harm and boost security. Node classification (XIAO et al., 2021) is a fundamental technique in graph analy-

sis with promising cybersecurity applications. Labeling nodes (e.g., devices, users) within a network as “normal"or “malicious"based on their behavior and connections allows us to identify and isolate suspicious nodes, effectively preventing or blocking ongoing attacks such as APTs. This enables security professionals to gain valuable insights into the behavior of entities, allowing them to identify potential threats and mitigate risks.

This chapter explores the application of efficient Graph Transformers for node classification in the context of anomaly detection and cybersecurity. We focus on recent advancements in attention mechanisms that aim to overcome the scalability challenges associated with traditional Graph Transformers (GTs). Our proposed GT model incorporates short-temporal dependencies through node features and contextual time embeddings by including edge features. Moreover, we further enhance its performance by employing a soft-attention mechanism to selectively prioritize the most representative nodes.

5.4 METHODOLOGY

Figura 18 – The flowchart of our proposal. Our approach explores efficient Graph Transformers (GT) for node classification in the context of anomaly detection and cybersecurity. Especially, the proposed model leverages temporal information from node and edge features through *Short-Term Embedding* and *Contextual Time Embedding* components while also employing a soft-attention mechanism to selectively prioritize the most relevant nodes. The *Message Passing* step incorporates recent advancements aimed at reducing computational complexity in graph tasks using GT.



In dynamic networks, modeling and predicting overall network behavior requires careful attention to both network topology and the chronological sequence of events. Our proposal introduces a methodology that addresses the temporal dimension through two complementary approaches. The first approach integrates temporal dependencies directly into node embed-

dings by embedding time-based features as node attributes, employing a pre-encoding technique we designate as *Short-Term Embedding*. The second approach explicitly captures the temporal evolution of events through a contextual embedding process based on edge features, termed *Contextual Time Embedding*. Together, these approaches enable a nuanced representation of temporal dynamics, effectively capturing the network's dynamic behavior at multiple granular levels.

Firstly, let $\mathcal{G} = (\mathcal{N}, \mathcal{E})$ represent a graph where $\mathcal{N}$ is a set of nodes ($|\mathcal{N}| = N$) and $\mathcal{E} \subseteq \mathcal{N} \times \mathcal{N}$ is a set of edges. Each node $u \in \mathcal{N}$ has associated features $\mathbf{x}_u \in \mathbb{R}^D$ and a corresponding label y_u. We define an adjacency matrix $\mathbf{A} = \{a_{uv}\} \in \{0,1\}^{N \times N}$, where $a_{uv} = 1$ if the edge $(u, v) \in \mathcal{E}$ exists and $a_{uv} = 0$ otherwise. The objective is to develop a function for predicting node-level outcomes, which means estimating labels for unlabeled or new nodes within the graph (e.g., when a given input represents a normal or abnormal network behavior).

5.4.1 SHORT-TERM EMBEDDING VIA NODE FEATURES

Our approach first executes a short-term embedding (Figure 18) to learn a network representation from continuous-time dynamic networks through *temporal random walks* (NGUYEN et al., 2018). Our goal is to capture the essential temporal dependencies at the most detailed level while also addressing the issue of missing node features in the datasets we are evaluating.

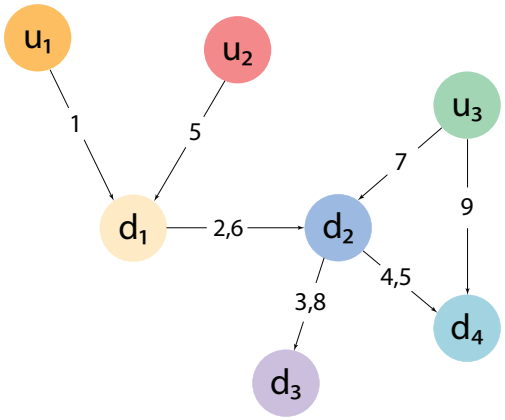
5.4.1.1 Temporal Random Walk

Temporal walks can capture valuable information about network interactions, including malicious activities like lateral movement. By modeling fine-grained temporal dependencies, temporal walks are particularly useful for detecting complex threats like APTs, enabling the tracking of attacker movements.

Methods that ignore time order are highly likely to miss crucial information and are prone to learning inappropriate node embeddings that do not accurately capture the dynamics in the network.

Specifically, to generate node features that are aware of time order, the input graph representing the events across the network is navigated using a temporal walk. A temporal walk explores a graph in ascending time order, capturing the essential temporal dependencies of the

Figura 19 – Understanding, modeling, and predicting network behavior accurately requires a clear grasp of the temporal sequence of events. In a dynamic graph with edges labeled by their arrival times, a sequence of nodes like $u_1 \to d_1 \to d_2 \to d_3$ constitutes a valid temporal walk, as it follows the edges chronologically. On the other hand, a typical random walk approach may mistakenly accept a sequence such as $u_3 \to d_2 \to d_4$, which fails to represent a legitimate temporal walk.



network at the finest granularity (e.g., at a time scale of seconds or milliseconds).

A temporal walk starting at node v_{i_1} and ending at node $v_{i_{L+1}}$ is represented by a series of edges $\{(v_{i_1}, v_{i_2}, t_{i_1}), (v_{i_2}, v_{i_3}, t_{i_2}), \ldots, (v_{i_L}, v_{i_{L+1}}, t_{i_L})\}$, where the edge timestamps satisfy the condition $t_{i_1} \leq t_{i_2} \leq \cdots \leq t_{i_L}$. In this context, a temporal walk denotes a time-respecting path in which edges are traversed in the non-decreasing order of their corresponding times.

In a computer network, the connections of a node may symbolize the interactions between users and devices, and the pattern of a temporal walk could provide substantial insights into a user's lateral movements within the network. Consider the sequence of events $e_1 = (u_1, d_1, 1)$, $e_2 = (d_1, d_2, 2)$, and $e_4 = (d_2, d_4, 4)$ depicted in Figure 19. If event e_1 represents user u_1 authenticating with device d_1 (e.g., a workstation, laptop, or smartphone) at time 1, and events e_2 and e_4 correspond to device d_1 communicating with d_2 at time 2 and d_2 communicating with d_4 at time 4, respectively, then the sequence $\{e_1, e_2, e_4\}$ may indicate the lateral movement of user u_1 within the network after authentication with device d_1.

The simple example illustrates the importance of modeling the actual sequence of events, such as user authentications within a computer network. Embedding methods that ignore time are prone to many issues, such as learning inappropriate node embeddings that do not accurately capture the dynamics in the network. For instance, the event sequence $\{e_7, e_4\}$ such that $e_7 = (u_3, d_2, 7)$ and $e_4 = (d_2, d_4, 4)$ can be considered a valid walk by using a general random walk but does not provide information on valid temporal events.

5.4.1.2 *Traversing the graph via a temporal walk to incorporate temporal dependencies in node features*

For a graph $\mathcal{G}$ with temporal information on its edges, we select the initial edge $e_t = (u, v, t) \in \mathcal{E}_T$, where $\mathcal{E}_T$ represents the set of temporal edges, using a uniform random distribution, where each edge has the same probability of being selected:

$$\mathbb{P}(e_t) = \frac{1}{|\mathcal{E}_T|} \tag{5.1}$$

Next, to start the temporal walk from the initial edge e_t at time t, we select the temporal neighbors of the outgoing node v, considering edges originating from v after time t. The temporal neighborhood of a node v at time t is denoted as $\Omega_t(v)$ and is defined as the set of nodes w' connected to v by an edge, with a timestamp t' greater than t:

$$\Omega_t(v) = \{(w', t') | (v, w', t') \in E \wedge t' > t\} \tag{5.2}$$

For example, suppose, from Figure 19 the initial edge is $(d_1, d_2, 2)$. The temporal neighborhood $\Omega_2(d_2)$ would consist of all edges originating from d_2 after time 2, resulting in the sequence $\{(d_3, 3), (d_4, 4), (d_4, 5), (d_3, 8)\}$.

It is important to note that the same node can appear multiple times in $\Omega_t(v)$ because multiple temporal interactions between two nodes can occur. For example, a user could have multiple authentication attempts on a host machine at different times. In contrast, a generic random walk without temporal consideration would only account for the static graph structure, resulting in $\Omega_t(d_2) = \{d_3, d_4\}$, leading to information loss.

The next node in the temporal walk $\mathcal{W}_T = \{(w^1, t^1), ..., (w^l, t^l)\}$, where l is the walk length, is chosen from $\Omega_t(v)$ using a uniform random distribution, as depicted in Equation 5.3:

$$\mathbb{P}(w) = \frac{1}{|\Omega_t(v)|} \tag{5.3}$$

5.4.1.3 *Modeling Short-term Temporal Dependencies in Temporal Random Walks*

To capture both the network's structure and its short-term temporal dynamics, a skip-gram model (MIKOLOV et al., 2013a) is employed. This approach, successfully utilized in graph embedding techniques such as CTDNE (NGUYEN et al., 2018) and recent network anomaly

detectors like Pikachu (PAUDEL; HUANG, 2022), optimizes the temporal order-preserving node embeddings. Specifically, we aim to maximize the log-probability of observing a temporal walk $\mathcal{W}_T$ for a node v, conditioned on its embedding, represented by the function $f : v \to \mathbb{R}^d$.

$$\max_f \sum_{v_j \in V_T} \log \mathbb{P}(\mathcal{W}_T | f(v_j)) \tag{5.4}$$

The optimization problem is made tractable by assuming the conditional independence of the node in the temporal walk $\mathcal{W}_T$ when observed with respect to the source node v:

$$\mathbb{P}(\mathcal{W}_T | f(v_j)) = \prod_{w^i \in \mathcal{W}_T} \mathbb{P}(w^i | f(v_j)) \tag{5.5}$$

The conditional likelihood of each source-neighborhood node pair can then be modeled using a softmax unit parametrized by a dot product of their embedding vectors.

$$\mathbb{P}(w^i | f(v_j)) = \frac{\exp(f(w^i) \cdot f(v_j))}{\sum_{v_k \in V_T} \exp(f(v_k) \cdot f(v_j))} \tag{5.6}$$

Finally, the optimization objective in Equation 5.4 simplifies to:

$$\max_f \sum_{v_j \in \mathcal{V}_T} \left[-\log Z_{v_j} + \sum_{w^i \in \mathcal{W}_T} f(w^i).f(v_j) \right] \tag{5.7}$$

where $Z_{v_j} = \sum_{v_k \in V_T} \exp(f(v_j) \cdot f(v_k))$ is a per-node partition function and can be approximated using negative sampling. The Skip-gram model will generate the node embedding $\mathbf{X}$ of the graph by encoding spatial information and short-term temporal information.

The time complexity for *Short-Term Embedding* is $\mathcal{O}(M + N(R \log M + RL\Delta + D))$, where $M = |\mathcal{E}_T|$ represent the number of temporal edges in the graph, N denotes the number of nodes, D represents the embedding dimension, R represents the number of temporal walks per node, L the maximum length of a temporal random walk, and Δ indicates the maximum degree of a node. Essentially, the *Short-Term Embedding* method exhibits a time complexity comparable to that presented by authors in (NGUYEN et al., 2018).

In particular, we highlight a linear relationship between the algorithm's runtime and the graph's number of edges and nodes. Furthermore, the runtime is sensitive to parameters related to the temporal walk configuration, exhibiting linear dependence on the number of walks per node, the maximum walk length, and the embedding dimension. Consequently, the algorithm's efficiency is particularly susceptible to the size and density of the graph, as well as

the dimensionality of the embeddings. Large, dense graphs and high-dimensional embeddings will inevitably lead to increased processing time.

5.4.2 MESSAGE PASSING: ACHIEVING EFFICIENT MESSAGE PASSING THROUGH A KERNELIZED GUMBEL-SOFTMAX OPERATOR.

Our Graph Transformer model is inspired by the Transformer architecture (VASWANI et al., 2017) and incorporates recent advances aimed at reducing computational complexity in graph tasks, as seen in (WU et al., 2022; JANG; GU; POOLE, 2017). We begin by defining a full-graph attentive network. Assuming $\mathbf{z}_u^{(0)} = \mathbf{x}_u$ as the initial node representation, this structure is designed to estimate latent interactions between individual nodes within the graph and enable the dense message passing between all nodes, fostering a rich and interconnected representation of the network:

$$\tilde{a}_{uv}^{(l)} = \frac{\exp\left(\left(W_Q^{(l)}\mathbf{z}_u^{(l)}\right)^\top \left(W_K^{(l)}\mathbf{z}_v^{(l)}\right)\right)}{\sum_{w=1}^{N}\exp\left(\left(W_Q^{(l)}\mathbf{z}_u^{(l)}\right)^\top \left(W_K^{(l)}\mathbf{z}_w^{(l)}\right)\right)}, \quad \mathbf{z}_u^{(l+1)} = \sum_{v=1}^{N}\tilde{a}_{uv}^{(l)}\cdot\left(W_V^{(l)}\mathbf{z}_v^{(l)}\right) \tag{5.8}$$

Where $W_Q^{(l)}$, $W_K^{(l)}$, and $W_V^{(l)}$ are learnable parameters in l-th layer.

As previously discussed in the literature (WU et al., 2022), the attention weights $\tilde{a}_{uv}^{(l)}$ (as defined in Equation 5.8) can be used to create a categorical distribution for generating latent edges. We can effectively obtain its neighbors by sampling from this categorical distribution multiple times for each node. Unfortunately, the computational cost of updating node representations in a single layer using Equation 5.8 is prohibitively high, scaling quadratically with the number of nodes.

To accelerate the full-graph model, recent researchers resort to approximations using kernel functions and random features (WU et al., 2022), which enable us to rewrite Equation 5.8:

$$\begin{aligned}\mathbf{z}_u^{(l+1)} &= \sum_{v=1}^{N}\frac{\phi\left(W_Q^{(l)}\mathbf{z}_u^{(l)}\right)^\top \phi\left(W_K^{(l)}\mathbf{z}_v^{(l)}\right)}{\sum_{w=1}^{N}\phi\left(W_Q^{(l)}\mathbf{z}_u^{(l)}\right)^\top \phi\left(W_K^{(l)}\mathbf{z}_w^{(l)}\right)}\cdot W_V^{(l)}\mathbf{z}_v^{(l)} \\ &= \frac{\phi\left(W_Q^{(l)}\mathbf{z}_u^{(l)}\right)^\top \sum_{v=1}^{N}\phi\left(W_K^{(l)}\mathbf{z}_v^{(l)}\right)\cdot W_V^{(l)}\mathbf{z}_v^{(l)\top}}{\phi\left(W_Q^{(l)}\mathbf{z}_u^{(l)}\right)^\top \sum_{w=1}^{N}\phi\left(W_K^{(l)}\mathbf{z}_w^{(l)}\right)}\end{aligned} \tag{5.9}$$

where $\phi(\mathbf{a})^\top\phi(\mathbf{b}) \approx \langle\Phi(\mathbf{a}),\Phi(\mathbf{b})\rangle_{\mathcal{V}} = \kappa(\mathbf{a},\mathbf{b})$, with $\kappa(\cdot,\cdot) : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ being a positive-definite kernel measuring the pairwise similarity, and $\Phi : \mathbb{R}^d \to \mathcal{V}$ a basis function

with a high-dimensional space $\mathcal{V}$, where $\Phi : \mathbb{R}^d \to \mathbb{R}^m$ is a low-dimensional feature map with random transformation. Positive Random Features (PRF) represent a promising choice for the feature map ϕ, as explored by (CHOROMANSKI et al., 2021). Their work demonstrated the efficacy of PRF in efficiently training softmax-based linear Transformers.

For notational convenience, let us define $\mathbf{q}_u = W_Q^{(l)}\mathbf{z}_u^{(l)}$, $\mathbf{k}_v = W_K^{(l)}\mathbf{z}_v^{(l)}$, and $\mathbf{v}_v = W_V^{(l)}\mathbf{z}_v^{(l)}$. With these definitions, Equation 5.9 can be rewritten as follows:

$$\mathbf{z}_u^{(l+1)} = \sum_{v=1}^{N} \frac{\phi(\mathbf{q}_u)^\top \phi(\mathbf{k}_v)}{\sum_{w=1}^{N} \phi(\mathbf{q}_u)^\top \phi(\mathbf{k}_w)} \cdot \mathbf{v}_v = \frac{\phi(\mathbf{q}_u)^\top \sum_{v=1}^{N} \phi(\mathbf{k}_v) \cdot \mathbf{v}_v^\top}{\phi(\mathbf{q}_u)^\top \sum_{w=1}^{N} \phi(\mathbf{k}_w)} \tag{5.10}$$

This approximation enhances computational efficiency, allowing for linear complexity in full-graph message passing. Reducing the algorithmic complexity of structure learning to linear with respect to the number of nodes allows us to work with larger graphs, which is typically the case in cybersecurity. A key advantage of Equation 5.10 lies in its ability to share the two summations across all nodes (u). This means we compute them only once and reuse them for all subsequent nodes, leading to considerable time savings. Such modifications are critical for effectively scaling graph structure learning to handle large-scale datasets.

While Equation 5.8 can be used as a basis to assign categorical weights to edges, directly sampling edges using these weights through a categorical distribution introduces discontinuities that hinder backpropagation. To optimize discrete graph structures in a differentiable way, we need to reformulate categorical sampling as a differentiable operation. The Gumbel-Softmax technique addresses this need, as it acts as a reparameterization method to approximate the discretely sampled edges through continuous relaxation. (WU et al., 2022; JANG; GU; POOLE, 2017).

Crucially, the Gumbel-Softmax (JANG; GU; POOLE, 2017) uses the softmax function as a continuous approximation to argmax, enabling the use of standard backpropagation for training neural networks with categorical outputs by controlling the closeness to hard discrete samples (MADDISON; MNIH; TEH, 2017):

$$\mathbf{z}_u^{(l+1)} \approx \sum_{v=1}^{N} \frac{\exp\left((\mathbf{q}_u^\top \mathbf{k}_u + g_v)/\tau\right)}{\sum_{w=1}^{N} \exp\left((\mathbf{q}_u^\top \mathbf{k}_w + g_w)/\tau\right)} \cdot \mathbf{v}_u, \quad g_w \sim \text{Gumbel}(0,1) \tag{5.11}$$

where τ is a temperature coefficient and particular, g_v values are independent and identically distributed (i.i.d.) samples drawn from the $\mathrm{Gumbel}(0,1)$ distribution. This distribution was independently discovered by Maddison, Mnih e Teh (2017). To sample from the

$\mathrm{Gumbel}(0,1)$ distribution, we can use inverse transform sampling: first, draw u from a uniform distribution $\mathrm{Uniform}(0,1)$, and then compute $g = -\log(-\log(u))$.

The Gumbel distribution arises naturally in extreme value theory (PINHEIRO; AND, 2016), meaning it is useful when working with max operations. If we take independent Gumbel-distributed random variables G_k, their maximum follows a well-defined distribution. This makes it ideal for selecting the most probable category in a discrete distribution. However, the argmax function is not differentiable, so the softmax function is used as an approximation to allow backpropagation through discrete variables by making them continuous, enabling gradient-based optimization.

The Gumbel-Softmax distribution is smooth for temperatures $\tau > 0$, ensuring a well-defined gradient. When it comes to learning, there is a trade-off between small temperatures, where samples are close to one-hot encoding but the gradients have high variance, and large temperatures, where samples are smoother but the gradients have low variance.

As Equation 5.11 requires $\mathcal{O}(N^2)$ to compute embeddings for N nodes, the kernelized Gumbel-Softmax operator is then used, as depicted in the Equation 5.12:

$$\begin{aligned}\mathbf{z}_u^{(l+1)} &\approx \sum_{v=1}^{N} \frac{\phi(\mathbf{q}_u/\sqrt{\tau})^\top \phi(\mathbf{k}_v/\sqrt{\tau}) e^{g_v/\tau}}{\sum_{w=1}^{N} \phi(\mathbf{q}_u/\sqrt{\tau})^\top \phi(\mathbf{k}_w/\sqrt{\tau}) e^{g_w/\tau}} \cdot \mathbf{v}_v \\ &= \frac{\phi(\mathbf{q}_u/\sqrt{\tau})^\top \sum_{v=1}^{N} e^{g_v/\tau} \phi(\mathbf{k}_v/\sqrt{\tau}) \cdot \mathbf{v}_v^\top}{\phi(\mathbf{q}_u/\sqrt{\tau})^\top \sum_{w=1}^{N} e^{g_w/\tau} \phi(\mathbf{k}_w/\sqrt{\tau})}\end{aligned} \tag{5.12}$$

To achieve a more robust representation, the averaged results are aggregated across K samples for each node, per layer, as depicted in Equation 5.13. Thus, for each layer, we sample K times for each node, such that there will be K sampled neighbored nodes for each node u. In practice, Equation 5.13 achieves message passing over a sampled latent graph with linear complexity.

$$\mathbf{z}_u^{(l+1)} = \frac{1}{K} \sum_{k=1}^{K} \frac{\phi\left(\frac{\mathbf{q}_u}{\sqrt{\tau}}\right)^\top \sum_{v=1}^{N} e^{g_{kv}/\tau} \phi\left(\frac{\mathbf{k}_v}{\sqrt{\tau}}\right) \cdot \mathbf{v}_v^\top}{\phi\left(\frac{\mathbf{q}_u}{\sqrt{\tau}}\right)^\top \sum_{w=1}^{N} e^{g_{kw}/\tau} \phi\left(\frac{\mathbf{k}_w}{\sqrt{\tau}}\right)} \tag{5.13}$$

5.4.3 TIME EMBEDDING: ENCODING CONTEXTUAL INFORMATION ABOUT THE TIME-DOMAIN VIA EDGE FEATURES

While Equation 5.13 effectively captures spatial relationships between nodes, it neglects the crucial temporal dimension. We introduce a mechanism for incorporating temporal context

into the node representations to address this.

Specifically, we leverage edge features $\mathbf{X}_E$ to extract contextual information about the temporal characteristics of each interaction. This information is then embedded into the edge representation $\mathbf{E}_{uv}$. As depicted in Figure 18, we introduce a dedicated *contextual time embedder* for this purpose. By incorporating these contextual embeddings into Equation 5.13, we enrich the node representations with temporal information, enabling our model to effectively capture both spatial and temporal relationships within the graph via message-passing. We perform K sampling iterations per node and compute the average over the aggregated outcomes. Specifically, at each layer of the Graph Transformer, K neighboring nodes are sampled for every node u, resulting in K distinct neighbor selections per node at that layer:

$$\mathbf{z}_u^{(l+1)} = \frac{1}{K} \sum_{k=1}^{K} \frac{\phi\left(\frac{\mathbf{q}_u}{\sqrt{\tau}}\right)^\top \sum_{v=1}^{N} e^{g_{kv}/\tau} \phi\left(\frac{\mathbf{k}_v}{\sqrt{\tau}}\right) \cdot \mathbf{v}_v^\top}{\phi\left(\frac{\mathbf{q}_u}{\sqrt{\tau}}\right)^\top \sum_{w=1}^{N} e^{g_{kw}/\tau} \phi\left(\frac{\mathbf{k}_w}{\sqrt{\tau}}\right)} \odot \mathbf{E}_{uv} \tag{5.14}$$

5.4.4 SOFT-ATTENTION MECHANISM

To further refine our node representations, we introduce (Equation 5.15) a soft-attention mechanism that operates directly on node embeddings during the test. This mechanism selectively amplifies the contributions of more significant nodes, allowing the model to focus on crucial information for accurate node classification.

$$\begin{aligned} \mathbf{A} &= \mathrm{softmax}(\sum_{i=1}^{N} \mathbf{z}_{ui}^{l+1} \odot \mathbf{z}_{ui}^{l+1}) \\ \mathbf{Z} &= \mathbf{z}_u^{l+1} \odot \mathbf{A} \end{aligned} \tag{5.15}$$

5.5 EXPERIMENTAL SETUP AND DATASETS

The node classification models were trained using Binary Cross-Entropy (BCE) loss. For evaluation, the ROC-AUC metric was utilized. The AUC-ROC metric is particularly chosen for assessing node classification models because it is effective in managing class imbalance. This reliability makes AUC-ROC a suitable choice, especially in situations where the dataset is imbalanced, which is common in intrusion detection tasks.

All models were two-layer architectures. Each dataset was split into training, validation,

and testing sets with ratios of 50%, 25%, and 25%, respectively. Each model was trained five times, and the average performance across these runs was considered. The learning rate used for the CERT R5.2 dataset was 0.001, and for Pivoting, it was 0.001.

The datasets used to evaluate the models in the present chapter were CERT R5.2, whose details were previously discussed in Chapter 4, and Pivoting (APRUZZESE et al., 2020). The Pivoting dataset comprises network traffic data in the form of network flows collected from a large organization over a single workday. These flows represent internal-to-internal network communications within the monitored environment. It is labeled through a manual process verified by the authors. The dataset is provided as a compressed .tar.gz file of approximately 1.5GB. Upon extraction, it yields a single CSV file of approximately 6GB containing nearly 75 million network flows.

5.6 RESULTS AND DISCUSSIONS

Table 4 presents a summary of our results, where we compare the performance of our model with various other state-of-the-art GNN models on different datasets. In the “model"column, we highlight *Proposal* and *Proposal*$_{CONTEXTUAL}$, which refer to our model trained without contextual information and with contextual information provided via edge features, respectively. Both models utilize *soft-attention* during testing.

As shown in Table 4, our approach generally outperforms others, considering that the proposed model is more stable and achieves ROC-AUC results that surpass a significant portion of competitors. When not superior, it is at least competitive. Furthermore, our model exhibits greater stability, with lower results variability than its counterparts. In certain cases, the variability is substantially lower. Nonetheless, we observe that the use of contextual time embeddings enhances results but introduces variability. This opens the door for future research aimed at minimizing this variability.

Additionally, our experiments explored the impact of soft-attention on the quality of node embeddings under various scenarios, analyzing model performance when applied during both training and testing, only during training, only during testing, and without application. Our results showed that applying soft-attention during the testing phase yielded superior performance compared to applying it during training and testing or solely during training.

For example, consider the model performance when using *contextual time embeddings* via edge features, as illustrated in Figure 20 (left side). We observe that the model with soft-

Model	Pivoting	CERT r5.2
APPNP (GASTEIGER; BOJCHEVSKI; GÜNNEMANN, 2019)	0.3621 ± 2.5991	0.3363 ± 21.4327
GAT	0.7221 ± 6.2165	0.6698 ± 0.2903
GCNJK (XU et al., 2018)	0.9079 ± 1.1291	0.7335 ± 0.4385
GATJK (XU et al., 2018)	0.9506 ± 2.9972	0.6899 ± 2.8179
GCN	0.168 ± 4.7431	0.6152 ± 0.477
GPRGNN (CHIEN et al., 2021)	0.502 ± 0.5154	0.6765 ± 4.6965
H2GCN (ZHU et al., 2020)	0.9621 ± 2.9956	0.7632 ± 1.8408
SGC (WU et al., 2019)	0.2972 ± 24.6716	0.6923 ± 0.0175
MixHop (ABU-EL-HAIJA et al., 2019)	0.8407 ± 6.8044	0.737 ± 0.9206
Proposal	0.9146 ± 0.5782	0.7930 ± 0.0629
$\text{Proposal}_{\text{CONTEXTUAL}}$	0.9217 ± 1.8939	0.7951 ± 0.2030

Tabela 4 – Performance in (%) on AUC-ROC metric for datasets Pivoting and CERT r5.2. *Proposal* and $Proposal_{CONTEXTUAL}$ refer to our model trained without contextual information and with contextual information provided via edge features, respectively. For baseline models, we use the implementation provided by (WU et al., 2022)

Figura 20 – We compared the application of self-attention on node representations in different scenarios on CERT r5.2. These results are evaluated when the model is trained with and without edge features, respectively (left and right side of the figure). Note that using self-attention at test time yields the best results.

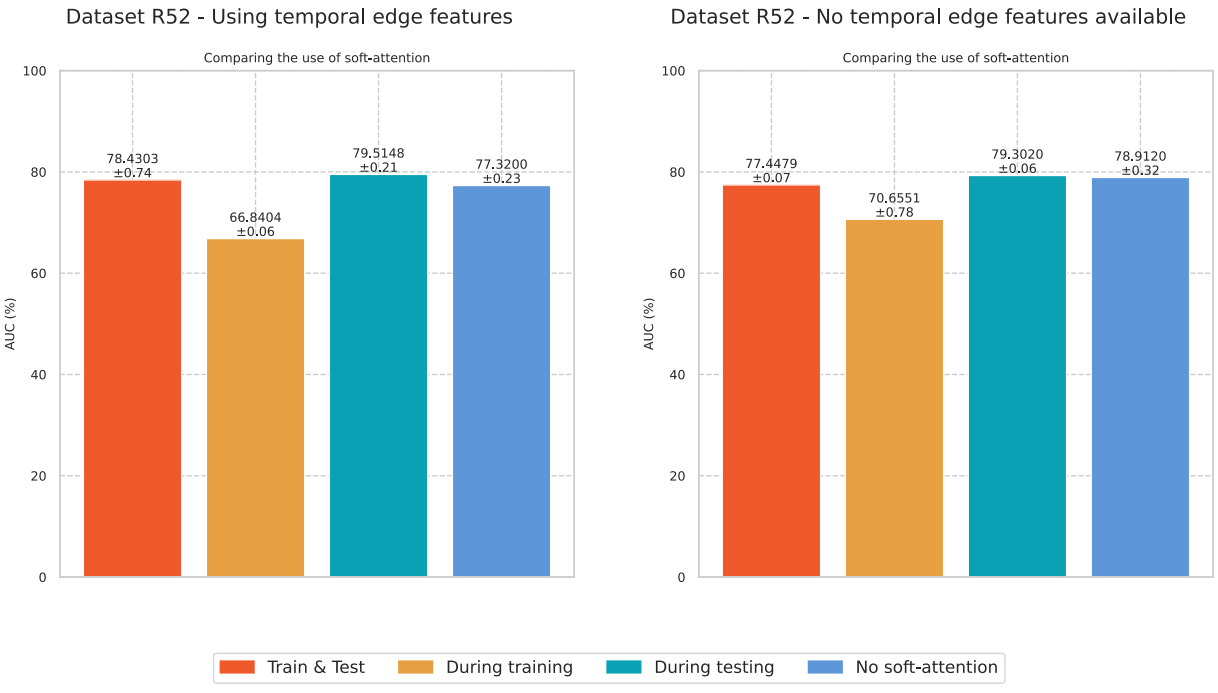
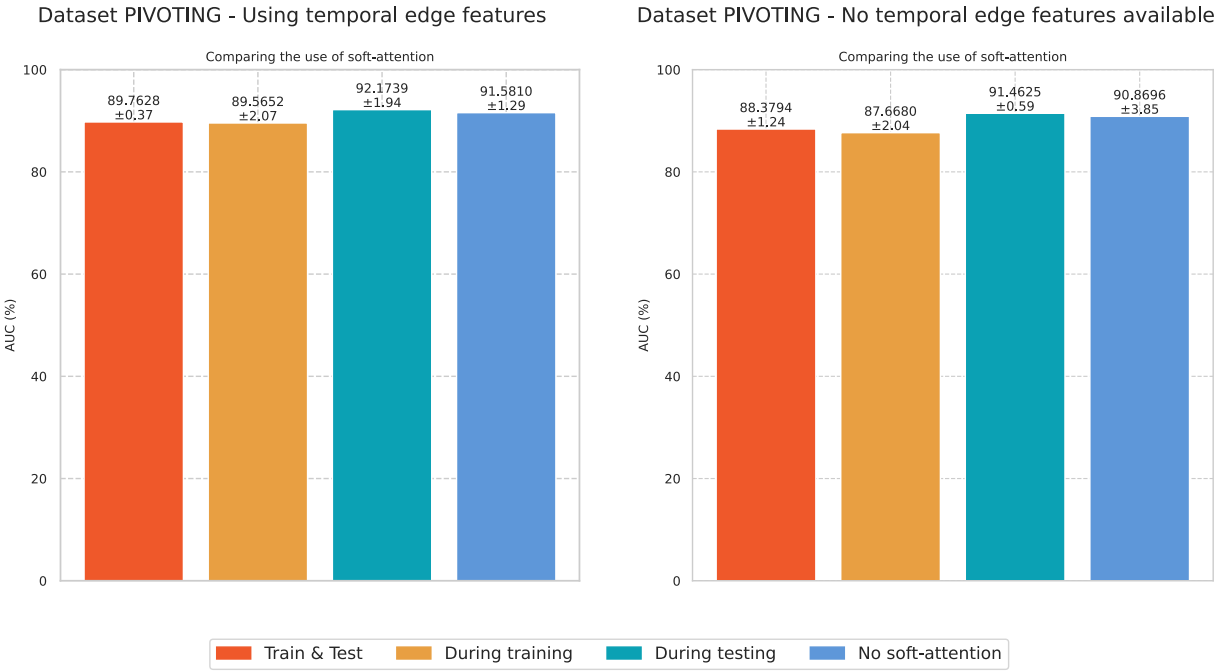


Figura 21 – Using the Pivoting dataset, we analyzed the use of self-attention on node representations across different scenarios. Once again, the results were evaluated based on whether the model was trained with or without contextual time embedding through edge features, as shown on the left and right sides of the figure, respectively. It's worth noting that applying self-attention during testing produces the best results.



attention during the testing phase (green bar) achieved a ROC-AUC of 79.5148 ± 0.21 on the CERT r5.2 dataset. In contrast, applying soft-attention during training & testing (orange bar) resulted in a ROC-AUC of 78.4303 ± 0.74, while not using soft-attention (blue bar) yielded 77.32 ± 0.23. This trend also holds when contextual time embeddings are not used, as shown in Figure 20 (right side).

This finding extends to other datasets, such as Pivoting in Figure 21, reinforcing the conclusion that soft-attention during testing is the most effective strategy. Conversely, applying soft-attention exclusively during training proved less efficient across all investigated scenarios.

5.7 MERITS OF THE PROPOSED APPROACH

5.7.1 ENABLE BOTH TOPOLOGICAL INFORMATION AND EDGE FEATURE WHILE SCALABLE

Firstly, our approach leverages the computational efficiency principles found in recent research on scalable Graph Transformer methods. Furthermore, unlike many existing methods, our model not only considers the graph topology but also learns representations from edge

features. This dual approach, incorporating both topological information and edge feature information, proves beneficial for intrusion detection via node classification.

By integrating these additional relational features, we surpass models relying solely on the node features, enabling more comprehensive and accurate intrusion detection.

5.7.2 APPLY SOFT-ATTENTION TO REINFORCE NODE SELECTION

Our approach employs a soft-attention mechanism applied to node representations, enabling the model to selectively prioritize relevant connections within the graph. This mechanism significantly diminishes the impact of less informative data, which frequently undermines the accuracy of node classification in graph transformer models.

This selective attention process helps to disentangle meaningful relationships from irrelevant noise, thereby enhancing the quality of the learned node representations and ultimately improving the accuracy of intrusion detection through node classification.

While existing GNNs struggle with large graphs and capturing long-range dependencies critical for identifying sophisticated attacks like APTs, Graph Transformers offer a solution by aggregating global information. However, the large contextual range of GT poses scalability challenges.

In this chapter, we explore efficient Graph Transformers for node classification in the context of anomaly detection and cybersecurity. In particular, we focus on recent advancements in attention mechanisms that aim to overcome the scalability challenges associated with traditional Graph Transformers (GTs). The proposed approach allows us to scale to large graphs with linear complexity concerning the number of nodes. As previously noted, the linear complexity allows us to work with bigger graphs, which is typically the case in the cybersecurity context.

Furthermore, we have noted that existing techniques that neglect time-based data analysis may miss significant details. As anomalies are often part of a sequence of events rather than isolated incidents, they require consideration of both spatial and temporal contexts for accurate identification. Thus, our model leverages temporal information from node and edge features while employing a soft-attention mechanism to selectively prioritize the most relevant nodes.

The experimental results have shown that our approach generally outperforms existing GNNs, considering that the proposed model is more stable and achieves ROC-AUC results that surpass a significant portion of competitors in the considered problem. When not superior, it is at least competitive. Furthermore, our model exhibits greater stability, with lower results

variability than its counterparts. In certain cases, the difference is substantial.

Moreover, the experimental results demonstrate that applying soft-attention exclusively during the testing phase significantly improves the quality metric compared to applying it during both training and testing, solely during training, or not using it at all. This finding highlights the effectiveness of soft-attention in selectively prioritizing relevant nodes during the evaluation process.

Furthermore, our model stands out from many existing methods by leveraging not only the graph's structure (topology) but also the information embedded within its connections (edge features). Combining topological and edge feature information, this dual approach significantly enhances intrusion detection accuracy through node classification. Nonetheless, we observe that using contextual time embeddings via edge features enhances results but introduces variability to some extent. This presents an opportunity for future research to explore ways to minimize this variability.

6 THESIS CONCLUSION

Highly sophisticated cyber-attacks constantly threaten the modern digital space by stealing sensitive information, leading to loss of privacy, confidential information, intellectual property, digital infrastructure, and revenue. Attackers update their knowledge constantly, developing new attack scenarios nearly daily. As Internet usage rises, cybersecurity companies must create more sophisticated security schemes.

Signature-based methods are widely employed in traditional IDSs in the industry due to their high efficiency and reliability in detecting known threats. These systems compare incoming data against a database of predefined signatures, triggering an alarm upon detecting a match. However, they are unable to identify attacks that do not align with known patterns, such as lateral movement, and prove ineffective against APT payloads that exploit zero-day vulnerabilities and leverage prior knowledge of the network environment.

On the other hand, Graph Machine Learning excels in this domain due to its capacity to discern complex, subtle patterns inherent in graph-structured data, such as those found in computer networks. By leveraging the natural graph representation of networks, these models can identify emerging threats that often evade traditional rule-based or signature-based methods. Moreover, Graph Machine Learning algorithms are adept at processing large volumes of interconnected data, enabling continuous adaptation to new malicious behaviors and enhancing the overall resilience of network security systems.

In this thesis, we approach the confluence of GNNs, Transformer architectures, and intrusion detection methodologies. Our primary focus is on cyber threats known as APTs, with particular emphasis on the critical Lateral Movement phase. To this end, we propose novel detection models based on GTs, a specialized GNN with the ability to capture long-range dependencies across the graph. Our first proposed model is designed to identify abnormal login attempts - critical indicators of lateral movement - and incorporates both soft-attention and gated mechanisms to selectively emphasize relevant node representations. Additionally, we introduce an innovative model for detecting suspicious endpoints, which integrates temporal dynamics to effectively capture the evolving nature of network activity and isolate compromised hosts.

The results indicate that the residual soft-attention aggregation method can produce more robust models that are less susceptible to the influence of less informative structural repre-

sentations of nodes. In contrast, methods such as sum and mean aggregation simply combine node representations without considering their relative importance, which can lead to the inclusion and propagation of less informative representations and negatively affect the model's performance. Specifically, the residual soft-attention aggregation method has been demonstrated to outperform competing approaches in terms of the false positive rate (FPR) within the proposed Gated-Graph Transform model. As previously mentioned, a low FPR is desirable, indicating that the method is less likely to generate false alarms. These findings address the first and second research questions presented in this thesis.

Concerning the third research question, a novel method was proposed for the detection of suspicious endpoints within a network through a node classification model. As demonstrated in this thesis, the model exhibits high accuracy and scalability, outperforming other evaluated GNN-based approaches. Furthermore, the applicability of this model is not limited to the detection of suspicious activities during APT scenarios; it can also be employed in broader operational contexts.

Regarding the incorporation of temporal dynamics, the proposed model employs two distinct architectural components designed to capture temporal information at different levels of granularity. One of these components is the *contextual time embedding*, which models temporal data through edge feature embeddings. However, experimental results indicated that this method does not significantly enhance the model's performance. These findings suggest that modeling temporal information through node features — as performed by the *short-time embedding* component — is sufficient on its own to achieve satisfactory results.

Nevertheless, there are various alternative strategies for integrating temporal dynamics into GNN-based architectures, which could be a focus for future research. One such strategies involves generating graph representations in an initial stage according to their temporal ordering using a GNN, followed by processing these representations with a sequential architecture such as an LSTM or a Transformer. Another possibility entails the direct integration of sequential mechanisms into the GNN architecture, thereby yielding a hybrid model capable of jointly capturing both structural and temporal dependencies. It is important to emphasize that these approaches open up a range of still underexplored possibilities, each posing its own methodological challenges and distinct levels of architectural complexity.

Notably, both the proposed models are complementary. While the link prediction model demonstrates robustness in detecting abnormal login attempts, the suspicious endpoint detection model provides an additional layer of protection by operating not only in the context of

APTs but also across a broader range of scenarios.

With regard to the use of the proposed approaches in intrusion detection systems, the complementary strengths and weaknesses of rule-based and machine learning-based intrusion detection systems motivate the pursuit of hybrid detection approaches. Despite their reliability in detecting known threats, rule-based systems are often perceived as more vulnerable to evasion attacks, given their reliance on predefined patterns. In contrast, machine learning-based tools, though more robust in detecting novel threats, are susceptible to data poisoning. In practice, these factors have led security professionals to favor machine learning-based tools that offer low false positive rates and can be deployed alongside rule-based systems. Such hybrid deployments aim to leverage the strengths of both paradigms, combining the precision and explainability of rule-based methods with the adaptability and pattern recognition capabilities of machine learning, to achieve more comprehensive and resilient intrusion detection.

Regarding the applicability of the proposed solutions in more diverse scenarios, we identify opportunities for its use in fraud detection systems in e-commerce, banking fraud detection, as well as in intrusion detection in vehicular networks. However, in the latter case, substantial challenges would arise due to the limited computational power of the devices operating within these networks.

It is important to note that while AI-based systems can analyze vast amounts of data, identify anomalies, and respond to threats faster than human analysts, they also introduce new vulnerabilities. Attackers can exploit biases and limitations in AI algorithms to deceive or manipulate these systems. This highlights the need for a deeper understanding of AI's vulnerabilities as well as the development of robust security measures for AI systems. Consequently, research on adversarial attack detection is critical, particularly with regard to future studies on adversarial attacks against GNN-based prediction systems. Moreover, a widely recognized limitation of machine learning models is their lack of interpretability, prompting ongoing research into explainable machine learning techniques to enhance transparency and trust in IDS. In this context, the exploration of explainability methods specifically tailored to GNNs represents a promising research direction.

6.1 PUBLISHED WORKS RESULTING FROM THIS THESIS

6.1.1 A SEMI-SUPERVISED AUTOENCODER APPROACH FOR EFFICIENT INTRUSION DETECTION IN NETWORK TRAFFIC

Published in *2023 IEEE Latin American Conference on Computational Intelligence (LA-CCI)*, this paper proposed an autoencoder-based solution trained in a semi-supervised manner to detect anomalies in network traffic. The autoencoder is trained on normal network traffic, allowing the model to learn a compact representation of the regular traffic data. The model then uses the reconstruction error as a mechanism for identifying anomalies in the network traffic and achieves strong performance in anomaly detection.

A. Almeida, L. Gonçalves, C. Zanchettin and B. L. D. Bezerra, "A Semi-Supervised Autoencoder Approach for Efficient Intrusion Detection in Network Traffic,"2023 IEEE Latin American Conference on Computational Intelligence (LA-CCI), Recife-Pe, Brazil, 2023, pp. 1-6, doi: 10.1109/LA-CCI58595.2023.10409491.

6.1.2 SG-RSRNN - SCORE GUIDED ROBUST SUBSPACE RECOVERY-BASED NEURAL NETWORK FOR NETWORK INTRUSION DETECTION

This work has been published in *2024 International Joint Conference on Neural Networks (IJCNN)*. The work introduces a new unsupervised deep learning approach for network intrusion detection using an autoencoder enhanced with a special regularizer for anomaly-robust feature extraction. Moreover, we incorporate a scoring network to guide anomaly score distribution, especially in the data's transition area, to better distinguish between normal and abnormal samples. The experimental results demonstrate significant improvement over existing methods on various datasets, showing the effectiveness of the proposed model in detecting network traffic anomalies.

*L. Gonçalves and C. Zanchettin, "SG-RSRNN - Score Guided Robust Subspace Recovery-based Neural Network for Network Intrusion Detection," 2024 International Joint Conference on Neural Networks (**IJCNN**), Yokohama, Japan, **2024**, pp. 1-8,*
doi: 10.1109/IJCNN60899.2024.10650391.

6.1.3 DETECTING ABNORMAL LOGINS BY DISCOVERING ANOMALOUS LINKS VIA GRAPH TRANSFORMERS

This work was recently published in the journal **Computers & Security**. This study proposes a novel approach: a residual soft-attention scheme that employs weighted sums for aggregation, resulting in improved node representations and better filtration of irrelevant information. Experimental results on three datasets confirm that this method excels at detecting abnormal authentications with fewer false positives than competitors.

Gonçalves, Luís, and Cleber Zanchettin. "Detecting abnormal logins by discovering anomalous links via graph transformers." ***Computers & Security*** *144 (**2024**): 103944.*

REFERÊNCIAS

ABU-EL-HAIJA, S.; PEROZZI, B.; KAPOOR, A.; ALIPOURFARD, N.; LERMAN, K.; HARUTYUNYAN, H.; STEEG, G. V.; GALSTYAN, A. MixHop: Higher-order graph convolutional architectures via sparsified neighborhood mixing. In: CHAUDHURI, K.; SALAKHUTDINOV, R. (Ed.). *Proceedings of the 36th International Conference on Machine Learning*. PMLR, 2019. (Proceedings of Machine Learning Research, v. 97), p. 21–29. Disponível em: <https://proceedings.mlr.press/v97/abu-el-haija19a.html>.

ALMEIDA, A.; GONçALVES, L.; ZANCHETTIN, C.; BEZERRA, B. L. D. A semi-supervised autoencoder approach for efficient intrusion detection in network traffic. In: *2023 IEEE Latin American Conference on Computational Intelligence (LA-CCI)*. [S.l.: s.n.], 2023. p. 1–6.

ALSHAMRANI, A.; MYNENI, S.; CHOWDHARY, A.; HUANG, D. A Survey on Advanced Persistent Threats: Techniques, Solutions, Challenges, and Research Opportunities. *IEEE Communications Surveys & Tutorials*, v. 21, n. 2, p. 1851–1877, 2019.

AMIN, M. A. R. A.; SHETTY, S.; NJILLA, L.; TOSH, D. K.; KAMHOUA, C. Hidden markov model and cyber deception for the prevention of adversarial lateral movement. *IEEE Access*, v. 9, p. 49662–49682, 2021.

APRUZZESE, G.; PIERAZZI, F.; COLAJANNI, M.; MARCHETTI, M. Detection and threat prioritization of pivoting attacks in large networks. *IEEE Transactions on Emerging Topics in Computing*, v. 8, n. 2, p. 404–415, 2020.

ATT&CK, M. Mitre att&ck. *URL: https://attack.mitre.org*, 2020.

ATT&CK®, M. *Lateral Movement, Tactic TA0008 - Enterprise | MITRE ATT&CK®*. 2018. Disponível em: <https://attack.mitre.org/tactics/TA0008/>.

BA, J. L. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

BAI, T.; BIAN, H.; DAYA, A. A.; SALAHUDDIN, M. A.; LIMAM, N.; BOUTABA, R. A machine learning approach for rdp-based lateral movement detection. In: *2019 IEEE 44th Conference on Local Computer Networks (LCN)*. [S.l.: s.n.], 2019. p. 242–245.

BIAN, H.; BAI, T.; SALAHUDDIN, M. A.; LIMAM, N.; DAYA, A. A.; BOUTABA, R. Uncovering lateral movement using authentication logs. *IEEE Transactions on Network and Service Management*, v. 18, n. 1, p. 1049–1063, 2021.

BOHARA, A.; NOUREDDINE, M. A.; FAWAZ, A.; SANDERS, W. H. An unsupervised multi-detector approach for identifying malicious lateral movement. In: *2017 IEEE 36th Symposium on Reliable Distributed Systems (SRDS)*. [S.l.: s.n.], 2017. p. 224–233.

BOWMAN, B.; HUANG, H. H. Towards next-generation cybersecurity with graph ai. *SIGOPS Oper. Syst. Rev.*, Association for Computing Machinery, New York, NY, USA, v. 55, n. 1, p. 61–67, jun. 2021. ISSN 0163-5980. Disponível em: <https://doi.org/10.1145/3469379.3469386>.

BOWMAN, B.; LAPRADE, C.; JI, Y.; HUANG, H. H. Detecting lateral movement in enterprise computer networks with unsupervised graph AI. In: *23rd International Symposium on Research in Attacks, Intrusions and Defenses (RAID 2020)*. San Sebastian:

USENIX Association, 2020. p. 257–268. ISBN 978-1-939133-18-2. Disponível em: <https://www.usenix.org/conference/raid2020/presentation/bowman>.

BRESSON, X.; LAURENT, T. *An Experimental Study of Neural Networks for Variable Graphs*. 2018. Disponível em: <https://openreview.net/forum?id=SJexcZc8G>.

CANARY, R. *Deep Dive: Lateral Movement.* 2020. <https://redcanary.com/blog/free-webinars/?wchannelid=bjg0ngnz0g&wmediaid=c6u20g49mv>. Accessed: 2021-03-23.

CAO, S.; SUN, X.; BO, L.; WEI, Y.; LI, B. BGNN4VD: Constructing Bidirectional Graph Neural-Network for Vulnerability Detection. *Information and Software Technology*, v. 136, p. 106576, 2021. ISSN 0950-5849. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0950584921000586>.

CARION, N.; MASSA, F.; SYNNAEVE, G.; USUNIER, N.; KIRILLOV, A.; ZAGORUYKO, S. End-to-end object detection with transformers. In: VEDALDI, A.; BISCHOF, H.; BROX, T.; FRAHM, J.-M. (Ed.). *Computer Vision – ECCV 2020*. Cham: Springer International Publishing, 2020. p. 213–229. ISBN 978-3-030-58452-8.

CASANOVA, P. V. G. C. A.; LIO, A. R. P.; BENGIO, Y. Graph attention networks. *ICLR. Petar Velickovic Guillem Cucurull Arantxa Casanova Adriana Romero Pietro Liò and Yoshua Bengio*, 2018.

CAVILLE, E.; LO, W. W.; LAYEGHY, S.; PORTMANN, M. Anomal-e: A self-supervised network intrusion detection system based on graph neural networks. *Knowledge-Based Systems*, Elsevier, v. 258, p. 110030, 2022.

CHAVEZ, M. R.; BUTLER, T. S.; REKAWEK, P.; HEO, H.; KINZLER, W. L. Chat Generative Pre-trained Transformer: why we should embrace this technology. *American Journal of Obstetrics and Gynecology*, 2023. ISSN 0002-9378. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0002937823001552>.

CHEN, C.; TAO, C.; WONG, N. Litegt: Efficient and lightweight graph transformers. In: *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*. New York, NY, USA: Association for Computing Machinery, 2021. (CIKM '21), p. 161–170. ISBN 9781450384469. Disponível em: <https://doi.org/10.1145/3459637.3482272>.

CHEN, D.; LIN, Y.; LI, W.; LI, P.; ZHOU, J.; SUN, X. Measuring and relieving the over-smoothing problem for graph neural networks from the topological view. *Proceedings of the AAAI Conference on Artificial Intelligence*, v. 34, n. 04, p. 3438–3445, Apr. 2020. Disponível em: <https://ojs.aaai.org/index.php/AAAI/article/view/5747>.

CHEN, D.; O'BRAY, L.; BORGWARDT, K. Structure-aware transformer for graph representation learning. In: CHAUDHURI, K.; JEGELKA, S.; SONG, L.; SZEPESVARI, C.; NIU, G.; SABATO, S. (Ed.). *Proceedings of the 39th International Conference on Machine Learning*. PMLR, 2022. (Proceedings of Machine Learning Research, v. 162), p. 3469–3489. Disponível em: <https://proceedings.mlr.press/v162/chen22r.html>.

CHEN, H.; WANG, L.; LIN, Y.; YEH, C.-C. M.; WANG, F.; YANG, H. Structured graph convolutional networks with stochastic masks for recommender systems. In: *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*. New York, NY, USA: Association for Computing

Machinery, 2021. (SIGIR '21), p. 614–623. ISBN 9781450380379. Disponível em: <https://doi.org/10.1145/3404835.3462868>.

CHEN, H.; YEH, C.-C. M.; WANG, F.; YANG, H. Graph neural transport networks with non-local attentions for recommender systems. In: *Proceedings of the ACM Web Conference 2022*. New York, NY, USA: Association for Computing Machinery, 2022. (WWW '22), p. 1955–1964. ISBN 9781450390965. Disponível em: <https://doi.org/10.1145/3485447.3512162>.

CHEN, M.; WEI, Z.; HUANG, Z.; DING, B.; LI, Y. Simple and Deep Graph Convolutional Networks. In: III, H. D.; SINGH, A. (Ed.). *Proceedings of the 37th International Conference on Machine Learning*. PMLR, 2020. (Proceedings of Machine Learning Research, v. 119), p. 1725–1735. Disponível em: <https://proceedings.mlr.press/v119/chen20v.html>.

CHEN, M.; YAO, Y.; LIU, J.; JIANG, B.; SU, L.; LU, Z. A novel approach for identifying lateral movement attacks based on network embedding. In: *2018 IEEE Intl Conf on Parallel Distributed Processing with Applications, Ubiquitous Computing Communications, Big Data Cloud Computing, Social Computing Networking, Sustainable Computing Communications (ISPA/IUCC/BDCloud/SocialCom/SustainCom)*. [S.l.: s.n.], 2018. p. 708–715.

CHEN, P.; DESMET, L.; HUYGENS, C. A study on advanced persistent threats. In: DECKER, B. D.; ZÚQUETE, A. (Ed.). *Communications and Multimedia Security*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2014. p. 63–72. ISBN 978-3-662-44885-4.

CHIEN, E.; PENG, J.; LI, P.; MILENKOVIC, O. Adaptive universal generalized pagerank graph neural network. In: *International Conference on Learning Representations*. [s.n.], 2021. Disponível em: <https://openreview.net/forum?id=n6jl7fLxrP>.

CHOROMANSKI, K. M.; LIKHOSHERSTOV, V.; DOHAN, D.; SONG, X.; GANE, A.; SARLOS, T.; HAWKINS, P.; DAVIS, J. Q.; MOHIUDDIN, A.; KAISER, L.; BELANGER, D. B.; COLWELL, L. J.; WELLER, A. Rethinking attention with performers. In: *International Conference on Learning Representations*. [s.n.], 2021. Disponível em: <https://openreview.net/forum?id=Ua6zuk0WRH>.

DAUD, N. N.; Ab Hamid, S. H.; SAADOON, M.; SAHRAN, F.; ANUAR, N. B. Applications of link prediction in social networks: A review. *Journal of Network and Computer Applications*, v. 166, p. 102716, 2020. ISSN 1084-8045. Disponível em: <https://www.sciencedirect.com/science/article/pii/S1084804520301909>.

DAUPHIN, Y. N.; FAN, A.; AULI, M.; GRANGIER, D. Language modeling with gated convolutional networks. In: PRECUP, D.; TEH, Y. W. (Ed.). *Proceedings of the 34th International Conference on Machine Learning*. PMLR, 2017. (Proceedings of Machine Learning Research, v. 70), p. 933–941. Disponível em: <https://proceedings.mlr.press/v70/dauphin17a.html>.

DEFFERRARD, M.; BRESSON, X.; VANDERGHEYNST, P. Convolutional neural networks on graphs with fast localized spectral filtering. In: *Proceedings of the 30th International Conference on Neural Information Processing Systems*. Red Hook, NY, USA: Curran Associates Inc., 2016. (NIPS'16), p. 3844–3852. ISBN 9781510838819.

DIN, A. M. ud; QURESHI, S. Limits of depth: Over-smoothing and over-squashing in gnns. *Big Data Mining and Analytics*, v. 7, n. 1, p. 205–216, 2024.

DIVAKARAN, A.; MOHAN, A. Temporal link prediction: A survey. *New Generation Computing*, Springer, v. 38, p. 213–258, 2020.

DOSOVITSKIY, A.; BEYER, L.; KOLESNIKOV, A.; WEISSENBORN, D.; ZHAI, X.; UNTERTHINER, T.; DEHGHANI, M.; MINDERER, M.; HEIGOLD, G.; GELLY, S.; USZKOREIT, J.; HOULSBY, N. An image is worth 16x16 words: Transformers for image recognition at scale. *ICLR*, 2021.

DOU, Y.; LIU, Z.; SUN, L.; DENG, Y.; PENG, H.; YU, P. S. Enhancing graph neural network-based fraud detectors against camouflaged fraudsters. In: *Proceedings of the 29th ACM International Conference on Information and Knowledge Management (CIKM'20)*. [S.l.: s.n.], 2020.

DUAN, D.; TONG, L.; LI, Y.; LU, J.; SHI, L.; ZHANG, C. Aane: Anomaly aware network embedding for anomalous link detection. In: *2020 IEEE International Conference on Data Mining (ICDM)*. [S.l.: s.n.], 2020. p. 1002–1007.

DWIVEDI, V. P.; BRESSON, X. A generalization of transformer networks to graphs. *AAAI Workshop on Deep Learning on Graphs: Methods and Applications*, 2021.

EKE, D. O. ChatGPT and the rise of generative AI: Threat to academic integrity? *Journal of Responsible Technology*, v. 13, p. 100060, 2023. ISSN 2666-6596. Disponível em: <https://www.sciencedirect.com/science/article/pii/S2666659623000033>.

FANG, Y.; WANG, C.; FANG, Z.; HUANG, C. Lmtracker: Lateral movement path detection based on heterogeneous graph embedding. *Neurocomputing*, v. 474, p. 37–47, 2022. ISSN 0925-2312. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0925231221018646>.

GASTEIGER, J.; BOJCHEVSKI, A.; GÜNNEMANN, S. Predict then propagate: Graph neural networks meet personalized pagerank. In: *International Conference on Learning Representations (ICLR)*. [S.l.: s.n.], 2019.

GLASSER, J.; LINDAUER, B. Bridging the gap: A pragmatic approach to generating insider threat data. In: *2013 IEEE Security and Privacy Workshops*. [S.l.: s.n.], 2013. p. 98–104.

GONçALVES, L.; ZANCHETTIN, C. Detecting abnormal logins by discovering anomalous links via graph transformers. *Computers & Security*, v. 144, p. 103944, 2024. ISSN 0167-4048. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0167404824002499>.

GONçALVES, L.; ZANCHETTIN, C. Sg-rsrnn - score guided robust subspace recovery-based neural network for network intrusion detection. In: *2024 International Joint Conference on Neural Networks (IJCNN)*. [S.l.: s.n.], 2024. p. 1–8.

GROVER, A.; LESKOVEC, J. Node2vec: Scalable feature learning for networks. In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. New York, NY, USA: Association for Computing Machinery, 2016. (KDD '16), p. 855–864. ISBN 9781450342322. Disponível em: <https://doi.org/10.1145/2939672.2939754>.

HAJIRAMEZANALI, E.; HASANZADEH, A.; NARAYANAN, K.; DUFFIELD, N.; ZHOU, M.; QIAN, X. Variational graph recurrent neural networks. In: WALLACH, H.; LAROCHELLE, H.; BEYGELZIMER, A.; ALCHé-BUC, F. d'; FOX, E.; GARNETT,

R. (Ed.). *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2019. v. 32. Disponível em: <https://proceedings.neurips.cc/paper/2019/file/a6b8deb7798e7532ade2a8934477d3ce-Paper.pdf>.

HALBOUNI, A.; GUNAWAN, T. S.; HABAEBI, M. H.; HALBOUNI, M.; KARTIWI, M.; AHMAD, R. Machine learning and deep learning approaches for cybersecurity: A review. *IEEE Access*, IEEE, v. 10, p. 19572–19585, 2022.

HAMILTON, W.; YING, Z.; LESKOVEC, J. Inductive representation learning on large graphs. In: GUYON, I.; LUXBURG, U. V.; BENGIO, S.; WALLACH, H.; FERGUS, R.; VISHWANATHAN, S.; GARNETT, R. (Ed.). *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2017. v. 30. Disponível em: <https://proceedings.neurips.cc/paper/2017/file/5dd9db5e033da9c6fb5ba83c7a7ebea9-Paper.pdf>.

HAMILTON, W. L. Graph representation learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, Morgan and Claypool, v. 14, n. 3, p. 1–159, 2020.

HAMILTON, W. L.; YING, R.; LESKOVEC, J. Representation learning on graphs: Methods and applications. *arXiv preprint arXiv:1709.05584*, 2017.

HAN, K.; WANG, Y.; CHEN, H.; CHEN, X.; GUO, J.; LIU, Z.; TANG, Y.; XIAO, A.; XU, C.; XU, Y.; YANG, Z.; ZHANG, Y.; TAO, D. A survey on vision transformer. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 45, n. 1, p. 87–110, 2023.

HAN, X.; CUI, S.; LIU, S.; ZHANG, C.; JIANG, B.; LU, Z. Network intrusion detection based on n-gram frequency and time-aware transformer. *Computers & Security*, v. 128, p. 103171, 2023. ISSN 0167-4048. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0167404823000810>.

HAVIV, A.; RAM, O.; PRESS, O.; IZSAK, P.; LEVY, O. Transformer language models without positional encodings still learn positional information. In: *Findings of the Association for Computational Linguistics: EMNLP 2022*. Abu Dhabi, United Arab Emirates: Association for Computational Linguistics, 2022. p. 1382–1390. Disponível em: <https://aclanthology.org/2022.findings-emnlp.99>.

HE, W.; WU, Y.; LI, X. Attention mechanism for neural machine translation: A survey. In: *2021 IEEE 5th Information Technology,Networking,Electronic and Automation Control Conference (ITNEC)*. [S.l.: s.n.], 2021. v. 5, p. 1485–1489.

HENDERSON, J.; MOHAMMADSHAHI, A.; COMAN, A.; MICULICICH, L. Transformers as graph-to-graph models. In: ELAZAR, Y.; ETTINGER, A.; KASSNER, N.; RUDER, S.; SMITH, N. A. (Ed.). *Proceedings of the Big Picture Workshop*. Singapore: Association for Computational Linguistics, 2023. p. 93–107. Disponível em: <https://aclanthology.org/2023.bigpicture-1.8>.

HOANG, V. T.; JEON, H.-J.; YOU, E.-S.; YOON, Y.; JUNG, S.; LEE, O.-J. Graph representation learning and its applications: A survey. *Sensors*, v. 23, n. 8, 2023. ISSN 1424-8220. Disponível em: <https://www.mdpi.com/1424-8220/23/8/4168>.

HOCHREITER, S.; SCHMIDHUBER, J. Long Short-Term Memory. *Neural Computation*, v. 9, n. 8, p. 1735–1780, 11 1997. ISSN 0899-7667. Disponível em: <https://doi.org/10.1162/neco.1997.9.8.1735>.

HU, Z.; DONG, Y.; WANG, K.; SUN, Y. Heterogeneous Graph Transformer. In: *Proceedings of The Web Conference 2020*. New York, NY, USA: Association for Computing Machinery, 2020. (WWW '20), p. 2704–2710. ISBN 9781450370233. Disponível em: <https://doi.org/10.1145/3366423.3380027>.

HUANG, S.; LIU, Y.; FUNG, C.; HE, R.; ZHAO, Y.; YANG, H.; LUAN, Z. Hitanomaly: Hierarchical transformers for anomaly detection in system log. *IEEE Transactions on Network and Service Management*, v. 17, n. 4, p. 2064–2076, 2020.

HUBBALLI, N.; SURYANARAYANAN, V. False alarm minimization techniques in signature-based intrusion detection systems: A survey. *Computer Communications*, v. 49, p. 1–17, 2014. ISSN 0140-3664. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0140366414001480>.

Jaafer Al-Saraireh and Ala' Masarweh. A novel approach for detecting advanced persistent threats. *Egyptian Informatics Journal*, v. 23, n. 4, p. 45–55, 2022. ISSN 1110-8665. Disponível em: <https://www.sciencedirect.com/science/article/pii/S1110866522000470>.

JANG, E.; GU, S.; POOLE, B. Categorical reparameterization with gumbel-softmax. In: *International Conference on Learning Representations*. [s.n.], 2017. Disponível em: <https://openreview.net/forum?id=rkE3y85ee>.

Justin Gilmer; Samuel S. Schoenholz; Patrick F. Riley; Oriol Vinyals; George E. Dahl. Neural Message Passing for Quantum Chemistry. In: PRECUP, D.; TEH, Y. W. (Ed.). *Proceedings of the 34th International Conference on Machine Learning*. PMLR, 2017. (Proceedings of Machine Learning Research, v. 70), p. 1263–1272. Disponível em: <https://proceedings.mlr.press/v70/gilmer17a.html>.

KAMATH, U.; GRAHAM, K. L.; EMARA, W. Transformers: Basics and introduction. In: _____. *Transformers for Machine Learning: A Deep Dive*. 1st edition. ed. [S.l.]: Chapman and Hall/CRC, 2022. (9781003170082), p. 19–26.

KELESIS, D.; VOGIATZIS, D.; KATSIMPRAS, G.; FOTAKIS, D.; PALIOURAS, G. *REDUCING OVERSMOOTHING IN GRAPH NEURAL NETWORKS BY CHANGING THE ACTIVATION FUNCTION*. 2023. Disponível em: <https://openreview.net/forum?id=8CDeu0f4i2>.

KERIVEN, N. Not too little, not too much: a theoretical analysis of graph (over)smoothing. In: KOYEJO, S.; MOHAMED, S.; AGARWAL, A.; BELGRAVE, D.; CHO, K.; OH, A. (Ed.). *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2022. v. 35, p. 2268–2281. Disponível em: <https://proceedings.neurips.cc/paper_files/paper/2022/file/0f956ca6f667c62e0f71511773c86a59-Paper-Conference.pdf>.

KHALEEFA, E.; ABDULAH, D. Concept and difficulties of advanced persistent threats (APT): Survey. *International Journal of Nonlinear Analysis and Applications*, Semnan University, v. 13, n. 1, p. 4037–4052, 2022. ISSN 2008-6822. Disponível em: <https://ijnaa.semnan.ac.ir/article_6230.html>.

KIM, J.; KANG, H.; KANG, P. Time-series anomaly detection with stacked Transformer representations and 1D convolutional network. *Engineering Applications of Artificial Intelligence*, v. 120, p. 105964, 2023. ISSN 0952-1976. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0952197623001483>.

KIM, J.; NGUYEN, D.; MIN, S.; CHO, S.; LEE, M.; LEE, H.; HONG, S. Pure transformers are powerful graph learners. In: KOYEJO, S.; MOHAMED, S.; AGARWAL, A.; BELGRAVE, D.; CHO, K.; OH, A. (Ed.). *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2022. v. 35, p. 14582–14595. Disponível em: <https://proceedings.neurips.cc/paper_files/paper/2022/file/5d84236751fe6d25dc06db055a3180b0-Paper-Conference.pdf>.

KING, I. J.; HUANG, H. H. Euler: Detecting network lateral movement via scalable temporal link prediction. *ACM Trans. Priv. Secur.*, Association for Computing Machinery, New York, NY, USA, v. 26, n. 3, jun. 2023. ISSN 2471-2566. Disponível em: <https://doi.org/10.1145/3588771>.

KIPF, T. N.; WELLING, M. Variational graph auto-encoders. *arXiv preprint arXiv:1611.07308*, 2016.

KIPF, T. N.; WELLING, M. Semi-supervised classification with graph convolutional networks. In: *International Conference on Learning Representations (ICLR)*. [S.l.: s.n.], 2017.

KITAEV, N.; KAISER, L.; LEVSKAYA, A. Reformer: The efficient transformer. In: *International Conference on Learning Representations*. [S.l.: s.n.], 2019.

KOLLIAS, G.; KALANTZIS, V.; IDÉ, T.; LOZANO, A.; ABE, N. Directed graph auto-encoders. In: *Proceedings of the AAAI conference on artificial intelligence*. [S.l.: s.n.], 2022. v. 36, n. 7, p. 7211–7219.

KUMAR, A.; SINGH, S. S.; SINGH, K.; BISWAS, B. Link prediction techniques, applications, and performance: A survey. *Physica A: Statistical Mechanics and its Applications*, v. 553, p. 124289, 2020. ISSN 0378-4371. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0378437120300856>.

LAKHA, B.; MOUNT, S. L.; SERRA, E.; CUZZOCREA, A. Anomaly detection in cybersecurity events through graph neural network and transformer based model: A case study with beth dataset. In: *2022 IEEE International Conference on Big Data (Big Data)*. [S.l.: s.n.], 2022. p. 5756–5764.

LALLIE, H. S.; SHEPHERD, L. A.; NURSE, J. R.; EROLA, A.; EPIPHANIOU, G.; MAPLE, C.; BELLEKENS, X. Cyber security in the age of covid-19: A timeline and analysis of cyber-crime and cyber-attacks during the pandemic. *Computers & Security*, v. 105, p. 102248, 2021. ISSN 0167-4048. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0167404821000729>.

LAPRADE, C.; BOWMAN, B.; HUANG, H. H. Picodomain: a compact high-fidelity cybersecurity dataset. *arXiv preprint arXiv:2008.09192*, 2020.

LEMAN, A.; WEISFEILER, B. A reduction of a graph to a canonical form and an algebra arising during this reduction. *Nauchno-Technicheskaya Informatsiya*, v. 2, n. 9, p. 12–16, 1968.

LEMAY, A.; CALVET, J.; MENET, F.; FERNANDEZ, J. M. Survey of publicly available reports on advanced persistent threat actors. *Computers & Security*, v. 72, p. 26–59, 2018. ISSN 0167-4048. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0167404817301608>.

LI, J.; YANG, B.; DOU, Z.-Y.; WANG, X.; LYU, M. R.; TU, Z. Information aggregation for multi-head attention with routing-by-agreement. In: BURSTEIN, J.; DORAN, C.; SOLORIO, T. (Ed.). *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Minneapolis, Minnesota: Association for Computational Linguistics, 2019. p. 3566–3575. Disponível em: <https://aclanthology.org/N19-1359>.

LI, Y.; GU, C.; DULLIEN, T.; VINYALS, O.; KOHLI, P. Graph matching networks for learning the similarity of graph structured objects. In: CHAUDHURI, K.; SALAKHUTDINOV, R. (Ed.). *Proceedings of the 36th International Conference on Machine Learning*. PMLR, 2019. (Proceedings of Machine Learning Research, v. 97), p. 3835–3845. Disponível em: <https://proceedings.mlr.press/v97/li19d.html>.

LI, Y.; ZEMEL, R.; BROCKSCHMIDT, M.; TARLOW, D. Gated graph sequence neural networks. In: *Proceedings of ICLR'16*. [s.n.], 2016. Disponível em: <https://www.microsoft.com/en-us/research/publication/gated-graph-sequence-neural-networks/>.

LI, Z.; CHENG, X.; SUN, L.; ZHANG, J.; CHEN, B. A hierarchical approach for advanced persistent threat detection with attention-based graph neural networks. *Security and Communication Networks*, v. 2021, n. 1, p. 9961342, 2021. Disponível em: <https://onlinelibrary.wiley.com/doi/abs/10.1155/2021/9961342>.

LIN, T.; WANG, Y.; LIU, X.; QIU, X. A survey of transformers. *AI Open*, v. 3, p. 111–132, 2022. ISSN 2666-6510. Disponível em: <https://www.sciencedirect.com/science/article/pii/S2666651022000146>.

LIU, C.; ZHAN, Y.; MA, X.; DING, L.; TAO, D.; WU, J.; HU, W. Gapformer: Graph transformer with graph pooling for node classification. In: ELKIND, E. (Ed.). *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, IJCAI-23*. International Joint Conferences on Artificial Intelligence Organization, 2023. p. 2196–2205. Main Track. Disponível em: <https://doi.org/10.24963/ijcai.2023/244>.

LIU, C.; ZHAN, Y.; MA, X.; DING, L.; TAO, D.; WU, J.; HU, W.; DU, B. Exploring sparsity in graph transformers. *Neural Networks*, v. 174, p. 106265, 2024. ISSN 0893-6080. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0893608024001898>.

LIU, F.; WEN, Y.; WU, Y.; LIANG, S.; JIANG, X.; MENG, D. Mltracer: Malicious logins detection system via graph neural network. In: *2020 IEEE 19th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*. [S.l.: s.n.], 2020. p. 715–726.

LIU, Y.; AO, X.; QIN, Z.; CHI, J.; FENG, J.; YANG, H.; HE, Q. Pick and choose: A gnn-based imbalanced learning approach for fraud detection. In: *Proceedings of the Web Conference 2021*. New York, NY, USA: Association for Computing Machinery, 2021. (WWW '21), p. 3168–3177. ISBN 9781450383127. Disponível em: <https://doi.org/10.1145/3442381.3449989>.

LO, W. W.; LAYEGHY, S.; SARHAN, M.; GALLAGHER, M.; PORTMANN, M. E-graphsage: A graph neural network based intrusion detection system for iot. In: *NOMS 2022-2022 IEEE/IFIP Network Operations and Management Symposium*. [S.l.: s.n.], 2022. p. 1–9.

Lü, L.; ZHOU, T. Link prediction in complex networks: A survey. *Physica A: Statistical Mechanics and its Applications*, v. 390, n. 6, p. 1150–1170, 2011. ISSN 0378-4371. Disponível em: <https://www.sciencedirect.com/science/article/pii/S037843711000991X>.

MA, L.; LIN, C.; LIM, D.; ROMERO-SORIANO, A.; DOKANIA, P. K.; COATES, M.; TORR, P.; LIM, S.-N. Graph Inductive Biases in Transformers without Message Passing. In: PMLR. *International Conference on Machine Learning*. [S.l.], 2023. p. 23321–23337.

MADDISON, C.; MNIH, A.; TEH, Y. The concrete distribution: A continuous relaxation of discrete random variables. In: INTERNATIONAL CONFERENCE ON LEARNING REPRESENTATIONS. *Proceedings of the international conference on learning Representations*. [S.l.], 2017.

MARCHEGGIANI, D.; TITOV, I. Encoding sentences with graph convolutional networks for semantic role labeling. In: *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. Copenhagen, Denmark: Association for Computational Linguistics, 2017. p. 1506–1515. Disponível em: <https://aclanthology.org/D17-1159>.

MASDARI, M.; KHEZRI, H. A survey and taxonomy of the fuzzy signature-based intrusion detection systems. *Applied Soft Computing*, v. 92, p. 106301, 2020. ISSN 1568-4946. Disponível em: <https://www.sciencedirect.com/science/article/pii/S1568494620302416>.

MIKOLOV, T.; CHEN, K.; CORRADO, G.; DEAN, J. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

MIKOLOV, T.; SUTSKEVER, I.; CHEN, K.; CORRADO, G. S.; DEAN, J. Distributed representations of words and phrases and their compositionality. In: BURGES, C.; BOTTOU, L.; WELLING, M.; GHAHRAMANI, Z.; WEINBERGER, K. (Ed.). *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2013. v. 26. Disponível em: <https://proceedings.neurips.cc/paper/2013/file/9aa42b31882ec039965f3c4923ce901b-Paper.pdf>.

MINK, J.; BENKRAOUDA, H.; YANG, L.; CIPTADI, A.; AHMADZADEH, A.; VOTIPKA, D.; WANG, G. Everybody's got ml, tell me what else you have: Practitioners' perception of ml-based security tools and explanations. In: *2023 IEEE Symposium on Security and Privacy (SP)*. [S.l.: s.n.], 2023. p. 2068–2085.

MÜLLER, L.; GALKIN, M.; MORRIS, C.; RAMPÁŠEK, L. Attending to graph transformers. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. Disponível em: <https://openreview.net/forum?id=HhbqHBBrfZ>.

NASIRI, E.; BERAHMAND, K.; ROSTAMI, M.; DABIRI, M. A novel link prediction algorithm for protein-protein interaction networks by attributed graph embedding. *Computers in Biology and Medicine*, v. 137, p. 104772, 2021. ISSN 0010-4825. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0010482521005667>.

NGUYEN, G. H.; LEE, J. B.; ROSSI, R. A.; AHMED, N. K.; KOH, E.; KIM, S. Dynamic network embeddings: From random walks to temporal random walks. In: *2018 IEEE International Conference on Big Data (Big Data)*. [S.l.: s.n.], 2018. p. 1085–1092.

NGUYEN, K.; HIEU, N. M.; NGUYEN, V. D.; HO, N.; OSHER, S.; NGUYEN, T. M. Revisiting over-smoothing and over-squashing using ollivier-ricci curvature. In: KRAUSE, A.; BRUNSKILL, E.; CHO, K.; ENGELHARDT, B.; SABATO, S.; SCARLETT, J. (Ed.). *Proceedings of the 40th International Conference on Machine Learning*. PMLR, 2023. (Proceedings of Machine Learning Research, v. 202), p. 25956–25979. Disponível em: <https://proceedings.mlr.press/v202/nguyen23c.html>.

NGUYEN, V.-A.; NGUYEN, D. Q.; NGUYEN, V.; LE, T.; TRAN, Q. H.; PHUNG, D. Regvd: Revisiting graph neural networks for vulnerability detection. In: *Proceedings of the ACM/IEEE 44th International Conference on Software Engineering: Companion Proceedings*. New York, NY, USA: Association for Computing Machinery, 2022. (ICSE '22), p. 178–182. ISBN 9781450392235. Disponível em: <https://doi.org/10.1145/3510454.3516865>.

NICOLSON, A.; PALIWAL, K. K. Masked multi-head self-attention for causal speech enhancement. *Speech Communication*, v. 125, p. 80–96, 2020. ISSN 0167-6393. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0167639320302806>.

O'NEIL, C. *Weapons of Math Destruction - How Big Data Increases Inequality and Threatens Democracy*. New York: Crown, 2016.

OORD, A. van den; DIELEMAN, S.; ZEN, H.; SIMONYAN, K.; VINYALS, O.; GRAVES, A.; KALCHBRENNER, N.; SENIOR, A.; KAVUKCUOGLU, K. Wavenet: Agenerative model for raw audio. *arXiv preprint arXiv:1609.03499*, 2016.

OORD, A. van den; KALCHBRENNER, N.; ESPEHOLT, L.; KAVUKCUOGLU, k.; VINYALS, O.; GRAVES, A. Conditional image generation with pixelcnn decoders. In: LEE, D.; SUGIYAMA, M.; LUXBURG, U.; GUYON, I.; GARNETT, R. (Ed.). *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2016. v. 29. Disponível em: <https://proceedings.neurips.cc/paper_files/paper/2016/file/b1301141feffabac455e1f90a7de2054-Paper.pdf>.

PAUDEL, R.; HUANG, H. Pikachu: Temporal walk based dynamic graph embedding for network anomaly detection. In: *NOMS*. [S.l.: s.n.], 2022.

PEROZZI, B.; AL-RFOU, R.; SKIENA, S. Deepwalk: Online learning of social representations. In: *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. New York, NY, USA: Association for Computing Machinery, 2014. (KDD '14), p. 701–710. ISBN 9781450329569. Disponível em: <https://doi.org/10.1145/2623330.2623732>.

PINHEIRO, E. C.; AND, S. L. P. F. A comparative review of generalizations of the gumbel extreme value distribution with an application to wind speed data. *Journal of Statistical Computation and Simulation*, Taylor & Francis, v. 86, n. 11, p. 2241–2261, 2016. Disponível em: <https://doi.org/10.1080/00949655.2015.1107909>.

POWELL, B. A. Detecting malicious logins as graph anomalies. *Journal of Information Security and Applications*, v. 54, p. 102557, 2020. ISSN 2214-2126. Disponível em: <https://www.sciencedirect.com/science/article/pii/S2214212620301332>.

PRITEE, Z. T.; ANIK, M. H.; ALAM, S. B.; JIM, J. R.; KABIR, M. M.; MRIDHA, M. Machine learning and deep learning for user authentication and authorization in cybersecurity: A state-of-the-art review. *Computers & Security*, v. 140, p. 103747, 2024. ISSN 0167-4048. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0167404824000488>.

RAMPášEK, L.; GALKIN, M.; DWIVEDI, V. P.; LUU, A. T.; WOLF, G.; BEAINI, D. Recipe for a General, Powerful, Scalable Graph Transformer. *Advances in Neural Information Processing Systems*, v. 35, 2022.

SARKER, I. H. Machine learning for intelligent data analysis and automation in cybersecurity: current and future prospects. *Annals of Data Science*, Springer, v. 10, n. 6, p. 1473–1498, 2023.

SHARMA, A.; GUPTA, B. B.; SINGH, A. K.; SARASWAT, V. K. Advanced persistent threats (apt): evolution, anatomy, attribution and countermeasures. *Journal of Ambient Intelligence and Humanized Computing*, v. 14, n. 7, p. 9355–9381, 2023. ISSN 1868-5145. Disponível em: <https://doi.org/10.1007/s12652-023-04603-y>.

SHI, Y.; HUANG, Z.; FENG, S.; ZHONG, H.; WANG, W.; SUN, Y. Masked label prediction: Unified message passing model for semi-supervised classification. In: ZHOU, Z.-H. (Ed.). *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*. International Joint Conferences on Artificial Intelligence Organization, 2021. p. 1548–1554. Main Track. Disponível em: <https://doi.org/10.24963/ijcai.2021/214>.

SHIRZAD, H.; VELINGKER, A.; VENKATACHALAM, B.; SUTHERLAND, D. J.; SINOP, A. K. Exphormer: sparse transformers for graphs. In: *Proceedings of the 40th International Conference on Machine Learning*. [S.l.]: JMLR.org, 2023. (ICML'23).

Siva Kumar, A.; RAJA, S.; PRITHA, N.; RAVIRAJ, H.; Babitha Lincy, R.; Jency Rubia, J. An adaptive transformer model for anomaly detection in wireless sensor networks in real-time. *Measurement: Sensors*, v. 25, p. 100625, 2023. ISSN 2665-9174. Disponível em: <https://www.sciencedirect.com/science/article/pii/S2665917422002598>.

STEFFENS, T. *Attribution of Advanced Persistent Threats*. [S.l.]: Springer, 2020.

SUKHBAATAR, S.; SZLAM, a.; WESTON, J.; FERGUS, R. End-to-end memory networks. In: CORTES, C.; LAWRENCE, N.; LEE, D.; SUGIYAMA, M.; GARNETT, R. (Ed.). *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2015. v. 28. Disponível em: <https://proceedings.neurips.cc/paper_files/paper/2015/file/8fb21ee7a2207526da55a679f0332de2-Paper.pdf>.

SUN, X.; YANG, J. Hetglm: Lateral movement detection by discovering anomalous links with heterogeneous graph neural network. In: *2022 IEEE International Performance, Computing, and Communications Conference (IPCCC)*. [S.l.: s.n.], 2022. p. 404–411.

TANG, B.; WANG, J.; YU, Z.; CHEN, B.; GE, W.; YU, J.; LU, T. Advanced Persistent Threat intelligent profiling technique: A survey. *Computers and Electrical Engineering*, v. 103, p. 108261, 2022. ISSN 0045-7906. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0045790622004931>.

TANG, J.; QU, M.; WANG, M.; ZHANG, M.; YAN, J.; MEI, Q. Line: Large-scale information network embedding. In: *Proceedings of the 24th International Conference on World Wide Web*. Republic and Canton of Geneva, CHE: International World Wide Web Conferences Steering Committee, 2015. (WWW '15), p. 1067–1077. ISBN 9781450334693. Disponível em: <https://doi.org/10.1145/2736277.2741093>.

TAY, Y.; DEHGHANI, M.; BAHRI, D.; METZLER, D. Efficient transformers: A survey. *ACM Comput. Surv.*, Association for Computing Machinery, New York, NY, USA, v. 55, n. 6, dez. 2022. ISSN 0360-0300. Disponível em: <https://doi.org/10.1145/3530811>.

VASWANI, A.; SHAZEER, N.; PARMAR, N.; USZKOREIT, J.; JONES, L.; GOMEZ, A. N.; KAISER, L. u.; POLOSUKHIN, I. Attention is All you Need. In: GUYON, I.; LUXBURG, U. V.; BENGIO, S.; WALLACH, H.; FERGUS, R.; VISHWANATHAN, S.; GARNETT, R. (Ed.). *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2017. v. 30. Disponível em: <https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf>.

VELAZQUEZ, C. Detecting and preventing attacks earlier in the kill chain. *SANS Institute Infosec Reading Room*, p. 1–21, 2015.

VELIčKOVIć, P. Everything is connected: Graph neural networks. *Current Opinion in Structural Biology*, v. 79, p. 102538, 2023. ISSN 0959-440X. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0959440X2300012X>.

VUKALOVIć, J.; DELIJA, D. Advanced persistent threats - detection and defense. In: *2015 38th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*. [S.l.: s.n.], 2015. p. 1324–1330.

WANG, S.; YU, S. P. Graph neural networks in anomaly detection. In: WU, L.; CUI, P.; PEI, J.; ZHAO, L. (Ed.). *Graph Neural Networks: Foundations, Frontiers, and Applications*. Singapore: Springer Singapore, 2022. p. 557–578.

WANG, X.; ZHANG, M. How powerful are spectral graph neural networks. In: CHAUDHURI, K.; JEGELKA, S.; SONG, L.; SZEPESVARI, C.; NIU, G.; SABATO, S. (Ed.). *Proceedings of the 39th International Conference on Machine Learning*. PMLR, 2022. (Proceedings of Machine Learning Research, v. 162), p. 23341–23362. Disponível em: <https://proceedings.mlr.press/v162/wang22am.html>.

WU, F.; SOUZA, A.; ZHANG, T.; FIFTY, C.; YU, T.; WEINBERGER, K. Simplifying graph convolutional networks. In: CHAUDHURI, K.; SALAKHUTDINOV, R. (Ed.). *Proceedings of the 36th International Conference on Machine Learning*. PMLR, 2019. (Proceedings of Machine Learning Research, v. 97), p. 6861–6871. Disponível em: <https://proceedings.mlr.press/v97/wu19e.html>.

WU, Q.; ZHAO, W.; LI, Z.; WIPF, D. P.; YAN, J. Nodeformer: A scalable graph structure learning transformer for node classification. In: KOYEJO, S.; MOHAMED, S.; AGARWAL, A.; BELGRAVE, D.; CHO, K.; OH, A. (Ed.). *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2022. v. 35, p. 27387–27401. Disponível em: <https://proceedings.neurips.cc/paper_files/paper/2022/file/af790b7ae573771689438bbcfc5933fe-Paper-Conference.pdf>.

WU, Q.; ZHAO, W.; YANG, C.; ZHANG, H.; NIE, F.; JIANG, H.; BIAN, Y.; YAN, J. Sgformer: Simplifying and empowering transformers for large-graph representations. In: *Advances in Neural Information Processing Systems (NeurIPS)*. [S.l.: s.n.], 2023.

WU, Z.; PAN, S.; CHEN, F.; LONG, G.; ZHANG, C.; YU, P. S. A comprehensive survey on graph neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, v. 32, n. 1, p. 4–24, 2021.

XIAO, S.; WANG, S.; DAI, Y.; GUO, W. Graph neural networks in node classification: survey and evaluation. *Machine Vision and Applications*, v. 33, n. 1, p. 4, 2021. ISSN 1432-1769. Disponível em: <https://doi.org/10.1007/s00138-021-01251-0>.

XU, K.; HU, W.; LESKOVEC, J.; JEGELKA, S. How powerful are graph neural networks? In: *International Conference on Learning Representations*. [s.n.], 2019. Disponível em: <https://openreview.net/forum?id=ryGs6iA5Km>.

XU, K.; LI, C.; TIAN, Y.; SONOBE, T.; KAWARABAYASHI, K.-i.; JEGELKA, S. Representation learning on graphs with jumping knowledge networks. In: DY, J.; KRAUSE, A. (Ed.). *Proceedings of the 35th International Conference on Machine Learning*. PMLR, 2018. (Proceedings of Machine Learning Research, v. 80), p. 5453–5462. Disponível em: <https://proceedings.mlr.press/v80/xu18c.html>.

YANG, L.-X.; LI, P.; YANG, X.; XIANG, Y.; JIANG, F.; ZHOU, W. Effective quarantine and recovery scheme against advanced persistent threat. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, v. 51, n. 10, p. 5977–5991, 2021.

YANG, M.; SHEN, Y.; LI, R.; QI, H.; ZHANG, Q.; YIN, B. A new perspective on the effects of spectrum in graph neural networks. In: CHAUDHURI, K.; JEGELKA, S.; SONG, L.; SZEPESVARI, C.; NIU, G.; SABATO, S. (Ed.). *Proceedings of the 39th International Conference on Machine Learning*. PMLR, 2022. (Proceedings of Machine Learning Research, v. 162), p. 25261–25279. Disponível em: <https://proceedings.mlr.press/v162/yang22n.html>.

YING, C.; CAI, T.; LUO, S.; ZHENG, S.; KE, G.; HE, D.; SHEN, Y.; LIU, T.-Y. Do Transformers Really Perform Badly for Graph Representation? In: RANZATO, M.; BEYGELZIMER, A.; DAUPHIN, Y.; LIANG, P.; VAUGHAN, J. W. (Ed.). *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2021. v. 34, p. 28877–28888. Disponível em: <https://proceedings.neurips.cc/paper_files/paper/2021/file/f1c1592588411002af340cbaedd6fc33-Paper.pdf>.

ZHANG, M.; CHEN, Y. Link Prediction Based on Graph Neural Networks. In: BENGIO, S.; WALLACH, H.; LAROCHELLE, H.; GRAUMAN, K.; CESA-BIANCHI, N.; GARNETT, R. (Ed.). *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2018. v. 31. Disponível em: <https://proceedings.neurips.cc/paper/2018/file/53f0d7c537d99b3824f0f99d62ea2428-Paper.pdf>.

ZHANG, M.; CHEN, Y. Link prediction based on graph neural networks. In: BENGIO, S.; WALLACH, H.; LAROCHELLE, H.; GRAUMAN, K.; CESA-BIANCHI, N.; GARNETT, R. (Ed.). *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2018. v. 31. Disponível em: <https://proceedings.neurips.cc/paper_files/paper/2018/file/53f0d7c537d99b3824f0f99d62ea2428-Paper.pdf>.

ZHANG, M.; CUI, Z.; NEUMANN, M.; CHEN, Y. An end-to-end deep learning architecture for graph classification. *Proceedings of the AAAI Conference on Artificial Intelligence*, v. 32, n. 1, Apr. 2018. Disponível em: <https://ojs.aaai.org/index.php/AAAI/article/view/11782>.

ZHENG, S.; LU, J.; ZHAO, H.; ZHU, X.; LUO, Z.; WANG, Y.; FU, Y.; FENG, J.; XIANG, T.; TORR, P. H.; ZHANG, L. Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. [S.l.: s.n.], 2021. p. 6881–6890.

ZHOU, J.; CUI, G.; HU, S.; ZHANG, Z.; YANG, C.; LIU, Z.; WANG, L.; LI, C.; SUN, M. Graph neural networks: A review of methods and applications. *AI Open*, v. 1, p. 57–81, 2020. ISSN 2666-6510. Disponível em: <https://www.sciencedirect.com/science/article/pii/S2666651021000012>.

ZHOU, Y.; LIU, S.; SIOW, J.; DU, X.; LIU, Y. Devign: Effective vulnerability identification by learning comprehensive program semantics via graph neural networks. In: _____. *Proceedings of the 33rd International Conference on Neural Information Processing Systems*. Red Hook, NY, USA: Curran Associates Inc., 2019.

ZHOU, Y.; LIU, S.; SIOW, J.; DU, X.; LIU, Y. Devign: Effective vulnerability identification by learning comprehensive program semantics via graph neural networks. In: WALLACH, H.; LAROCHELLE, H.; BEYGELZIMER, A.; ALCHé-BUC, F. d'; FOX, E.; GARNETT, R. (Ed.). *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2019. v. 32. Disponível em: <https://proceedings.neurips.cc/paper_files/paper/2019/file/49265d2447bc3bbfe9e76306ce40a31f-Paper.pdf>.

ZHOU, Z.; LI, G.; WANG, G. A hybrid of transformer and CNN for efficient single image super-resolution via multi-level distillation. *Displays*, v. 76, p. 102352, 2023. ISSN 0141-9382. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0141938222001706>.

ZHU, H.; LUO, D.; TANG, X.; XU, J.; LIU, H.; WANG, S. Self-explainable graph neural networks for link prediction. *CoRR*, 2023.

ZHU, J.; YAN, Y.; ZHAO, L.; HEIMANN, M.; AKOGLU, L.; KOUTRA, D. Beyond homophily in graph neural networks: Current limitations and effective designs. *Advances in neural information processing systems*, v. 33, p. 7793–7804, 2020.

ZIMBA, A.; CHEN, H.; WANG, Z.; CHISHIMBA, M. Modeling and detection of the multi-stages of advanced persistent threats attacks based on semi-supervised learning and complex networks characteristics. *Future Generation Computer Systems*, v. 106, p. 501–517, 2020. ISSN 0167-739X. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0167739X19316267>.

ZOLA, F.; SEGUROLA-GIL, L.; BRUSE, J.; GALAR, M.; ORDUNA-URRUTIA, R. Network traffic analysis through node behaviour classification: a graph-based approach with temporal dissection and data-level preprocessing. *Computers & Security*, v. 115, p. 102632, 2022. ISSN 0167-4048. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0167404822000311>.