Recovering Missing Exploitation Knowledge Between Software Weaknesses and Attack Patterns

1st Anonymous

***Abstract*—Understanding how software weaknesses can be exploited is essential for vulnerability analysis, autonomous penetration testing, and LLM-based security agents. However, approximately 65% of Common Weakness Enumeration (CWE) entries lack a corresponding attack pattern (CAPEC). This substantial coverage gap limits access to structured exploitation knowledge and, consequently, constrains automated security reasoning.**

We show that CAPEC information substantially improves LLM-generated exploitation plans, highlighting the practical importance of recovering missing CWE–CAPEC relationships. Therefore, we formulate this problem as a heterogeneous link prediction problem and propose CC-GATNE. This framework jointly models vulnerabilities, weaknesses, attack patterns, and adversarial techniques within a unified graph representation.

CC-GATNE extends heterogeneous graph learning through cybersecurity-specific graph augmentation, semantic feature integration, attention-based aggregation, and relation-balanced context generation. Experimental results demonstrate substantial improvements over existing approaches. Compared with prior GNN-based methods, CC-GATNE improves AUC from 59.18% to 72.14%, while increasing F1 from 23.00% to 69.61% over the previous approaches and LLM-based baselines.

***Index Terms*—link prediction, weakness, exploitation, CWE, CAPEC**

1. Introduction

Software weaknesses and their exploitation mechanisms constitute a fundamental component of vulnerability analysis, secure software engineering, penetration testing, and cybersecurity education. More recently, they have also become increasingly important for autonomous security agents and large language model (LLM)-based systems that reason about vulnerabilities, generate attack procedures, or assist offensive and defensive security workflows. In these settings, understanding not only *what* is vulnerable, but also *how* a weakness can be exploited, is essential for effective security reasoning.

This relationship is primarily represented through the Common Weakness Enumeration (CWE) and the Common Attack Pattern Enumeration and Classification (CAPEC) frameworks. CWE defines classes of software weaknesses, while CAPEC describes attack patterns that exploit them. Together, these frameworks provide a natural bridge between vulnerability root cause and exploitation knowledge, enabling analysts and automated systems to connect vulnerable software behaviors with potential attack procedures. To better understand the practical importance of knowing which attack pattern is related to a given root cause, we investigate and show how CAPEC knowledge improves automated exploitation reasoning, as shown in Section 4.

However, despite their importance, the relationships between CWE and CAPEC remain substantially incomplete, with approximately 65% of the CWEs without a related CAPEC entry. As a consequence, many weaknesses remain disconnected from relevant exploitation knowledge, limiting the ability of both human analysts and automated systems to reason about how vulnerabilities may be operationalized in practice. Missing CWE–CAPEC links therefore represent more than an incomplete taxonomy: they correspond to missing procedural knowledge that can affect downstream tasks such as vulnerability analysis, exploitation planning, attack-path construction, and autonomous penetration testing.

Recovering missing CWE–CAPEC relationships is challenging. Existing approaches have primarily relied on textual similarity, keyword matching, or conventional machine learning techniques operating on isolated descriptions. While these methods can capture lexical relationships, they often fail to model the inherently relational nature of cybersecurity knowledge. In practice, exploitation relationships rarely emerge from textual similarity alone. Weaknesses, vulnerabilities, attack patterns, adversarial techniques, and procedural behaviors form a highly interconnected ecosystem in which meaningful relationships are expressed through structural dependencies rather than direct semantic overlap.

Our key insight is that missing exploitation knowledge can be recovered through heterogeneous cybersecurity reasoning. Rather than treating CWE and CAPEC entries as isolated textual artifacts, we model vulnerabilities, weaknesses, attack patterns, and adversarial techniques as components of a unified multi-relational knowledge graph. Within this representation, previously unseen CWE–CAPEC relationships can be inferred from structural dependencies spanning multiple cyber-

security frameworks, allowing the model to recover exploitation knowledge that is not explicitly encoded in existing mappings.

GNNs offer a natural way to model CWE–CAPEC prediction as a relational reasoning problem rather than as isolated text matching. Yet cybersecurity graphs pose specific challenges, including sparse supervision, imbalanced relation types, and noisy heterogeneous neighborhoods. We address these challenges by introducing **CC-GATNE**, a heterogeneous GNN framework for missing CWE–CAPEC link prediction. The framework combines enriched cybersecurity graph construction with semantic node features, attention-based aggregation, and layer-balanced random walks, enabling the model to capture both structural and exploitation-oriented relationships across CWEs, CAPECs, CVEs, and ATT&CK techniques. To support reproducibility and future research, we make the corresponding code and experimental pipeline publicly available[1].

In summary, this paper addresses the following research question:

> How can missing CWE–CAPEC relationships be recovered from existing cybersecurity knowledge?

We make the following contributions:

- We introduce CWE–CAPEC completion as a heterogeneous cybersecurity knowledge-recovery problem. Rather than treating the CWE–CAPEC mapping as a textual similarity between isolated descriptions, we formulate it as a link prediction problem over a multi-relational cybersecurity knowledge graph.
- We propose CC-GATNE, a heterogeneous graph-learning framework for recovering missing CWE–CAPEC relationships. CC-GATNE extends GATNE [1] with cybersecurity-specific graph augmentation, semantic node initialization, attention-based neighbor aggregation, and layer-balanced random walks to better capture sparse and imbalanced exploitation-related relations.
- We demonstrate the practical value of CWE–CAPEC completion for exploitation reasoning. Through an LLM-based exploitation-planning experiment, we show that CAPEC information improves generated attack plans, and that predicted CAPEC links can provide useful procedural context when explicit mappings are missing.

2. Background and Problem Formulation

2.1. Cybersecurity Knowledge Frameworks

Cybersecurity knowledge is commonly organized through structured taxonomies that describe vulnerabilities, weaknesses, attack patterns, and adversarial behaviors. Among the most widely adopted standards are the Common Weakness Enumeration (CWE) and the Common Attack Pattern Enumeration and Classification (CAPEC) frameworks, both maintained by MITRE.

CWE provides a hierarchical catalog of software weaknesses that highlight the root cause of security vulnerabilities. Each CWE entry describes a specific class of weakness, such as improper input validation, buffer overflows, or authentication flaws. This framework is extensively used in vulnerability classification, secure software development, and automated security analysis.

Complementary to CWE, CAPEC describes adversarial attack patterns that capture how attackers exploit weaknesses in real-world systems. CAPEC entries provide procedural and operational knowledge, including attack prerequisites, execution flows, required attacker skills, and potential consequences. While CWE focuses on *what is wrong* in a system, CAPEC focuses on *how an attacker can exploit it*.

2.2. Problem formulation

Our goal is to recover missing exploitation knowledge between software weaknesses and attack patterns. We formulate this task as link prediction over a heterogeneous cybersecurity graph. We represent cybersecurity knowledge as a heterogeneous graph

$$G = (V, E, \mathcal{T}_V, \mathcal{T}_E),$$

where V is the set of cybersecurity entities and E is the set of relationships among them. Each node $v \in V$ has a type $\tau(v) \in \mathcal{T}_V$, such as `CWE`, `CAPEC`. Each edge $e \in E$ has a relation type $\rho(e) \in \mathcal{T}_E$, such as `CWE--CWE`, `CWE--CAPEC`. This representation allows the graph to preserve both the semantic roles of cybersecurity entities and the relation types connecting them.

The target relation in this work is the CWE–CAPEC relationship: whether a weakness can be associated with an attack pattern that describes a plausible exploitation procedure. Given a CWE node $u \in$ `CWE` and a CAPEC node $v \in$ `CAPEC`, we learn a scoring function

$$f(u, v) \rightarrow [0, 1],$$

where higher scores indicate that v is more likely to represent a relevant attack pattern for exploiting weakness u. Candidate CAPEC entries can therefore be ranked for each CWE according to their predicted relationship score.

The task is to rank candidate CAPEC nodes for each CWE node according to their predicted CWE–CAPEC score. Higher-ranked CAPEC nodes are interpreted as more likely to provide relevant exploitation knowledge for the corresponding weakness. This formulation allows missing CWE–CAPEC relationships to be inferred from both textual node information and structural dependencies across vulnerabilities, weaknesses, attack patterns, and adversarial behaviors.

1. https://anonymous.4open.science/status/cwe_capec-4600

Additional background on graph neural networks and heterogeneous message passing is provided in Appendix 11.2.

3. Related Work

Automated mapping between cybersecurity knowledge bases has been widely studied, particularly for linking vulnerabilities to weakness classes [2]–[5] and threat intelligence text to adversarial tactics, techniques, and procedures [6]–[15]. These efforts demonstrate the importance of connecting cybersecurity artifacts across knowledge bases, but they primarily focus on vulnerability classification or operational adversary behavior rather than the relationship between weakness classes and exploitation procedures.

Compared with CVE-to-CWE and ATT&CK mapping, the specific problem of linking CWE weaknesses to CAPEC attack patterns has received less attention, despite its direct relevance to exploitation reasoning. Early work approached this space through structured knowledge representation. Andrei [16] proposed an OWL-based semantic model derived from CWE and CAPEC dictionaries, enabling automated classification and SPARQL-based querying over security concepts.

More recent work has studied textual and semantic similarity for identifying relationships between vulnerabilities, weaknesses, and attack patterns. Kanakogi et al. [17], [18] evaluated methods including TF-IDF, Doc2Vec, Universal Sentence Encoder, and Sentence-BERT for tracing relationships between cybersecurity descriptions. Their findings suggest that lexical signals can remain competitive in cybersecurity mapping tasks, where domain-specific terminology and sparse descriptions often limit the effectiveness of generic semantic embeddings. Bonomi et al. [19] further combine sentence embeddings with technical keyword analysis to better capture domain-specific terms and acronyms.

These approaches primarily operate over isolated textual descriptions. In contrast, our work models CWE–CAPEC completion as a heterogeneous graph reasoning problem, where exploitation relationships may be inferred not only from direct semantic similarity but also from structural dependencies across vulnerabilities, weaknesses, attack patterns, and adversarial techniques.

3.1. Graph-based reasoning over cybersecurity knowledge

Graph-based methods have also been proposed for reasoning over cybersecurity knowledge bases. Ren et al. [20] introduce a two-stage framework combining GCN-based classification with ConvE-based knowledge graph completion over CVE, CWE, and CAPEC data. Their approach uses textual features from related security entities and then performs link inference over constructed triples. We do not include this system as a baseline because the available description does not provide sufficient implementation detail to reproduce the pipeline or verify that its evaluation protocol is directly comparable to ours. In particular, the use of CAPEC-derived features before downstream link inference makes it difficult to isolate the CWE–CAPEC completion task under the leakage-prevention protocol used in this paper. Nevertheless, we provide an approximation of the work with a GCN comparison and report it in Appendix 8.

Our work also relates to graph representation learning methods for homogeneous, relational, and heterogeneous graphs. Homogeneous GNNs such as GCN and GAT aggregate neighborhood information but do not explicitly preserve node or edge heterogeneity [21], [22]. Relational and heterogeneous architectures, including R-GCN, HGT, HetGNN, and Simple-HGN, incorporate relation-specific or type-specific transformations to better model multi-relational graphs [23]–[26]. GATNE [1] is particularly relevant because it learns both shared and relation-specific node representations in attributed multiplex networks, making it suitable for graphs in which the same cybersecurity entity participates in multiple semantic relations. For completeness, Appendix 11.2 provides additional background on graph neural networks and heterogeneous message passing.

3.2. Positioning of this work

This work focuses on recovering missing exploitation knowledge between CWE weakness classes and CAPEC attack patterns. Unlike prior approaches that mainly rely on textual similarity, we formulate CWE–CAPEC completion as heterogeneous link prediction over a multi-relational cybersecurity graph.

We further introduce CC-GATNE, a cybersecurity-specific extension of GATNE that improves representation learning at both the data and model levels. Finally, we evaluate CWE–CAPEC completion not only as a link-prediction task, but also through its impact on LLM-based exploitation planning. This positions the task as practical knowledge recovery for automated security reasoning, rather than merely taxonomy matching.

4. Motivation

The incompleteness of CWE–CAPEC mappings is not merely a taxonomy-maintenance issue. CWE entries describe vulnerable software behaviors, while CAPEC entries describe attack patterns and exploitation procedures that adversaries may use against those weaknesses. Missing CWE–CAPEC links therefore correspond to missing exploitation knowledge: a weakness may be known, but how it can be exploited may remain disconnected from it.

This gap matters increasingly for automated security reasoning. LLM-based security agents, exploitation planners, and autonomous penetration-testing systems often transform vulnerability information into actionable attack plans. If CAPEC information provides procedural context that is absent from CVE or CWE descriptions alone, then missing CWE–CAPEC links can directly limit the quality of these systems.

The scale of the problem is substantial. In our dataset, 956 CWE entries and 615 CAPEC entries yield 587,940 candidate CWE–CAPEC pairs. Although many pairs are unrelated, identifying meaningful relationships requires domain expertise and many valid weakness–attack-pattern relationships may be indirect and difficult to identify from the CWE and CAPEC descriptions alone. Our approach is useful precisely because it exploits additional structural evidence from related cybersecurity entities, to infer missing links among mostly stable concepts.

4.1. Exploitation Planning

Before designing a model to recover missing CWE–CAPEC links, we first ask whether CAPEC information provides measurable value for downstream security reasoning. We therefore evaluate whether adding CAPEC attack-pattern information improves LLM-generated exploitation plans compared with using CVE or CVE+CWE information alone.

Specifically, we investigate the following research question:

> *Does CAPEC information help LLMs generate exploitation plans that better align with expert-grounded vulnerability walkthroughs?*

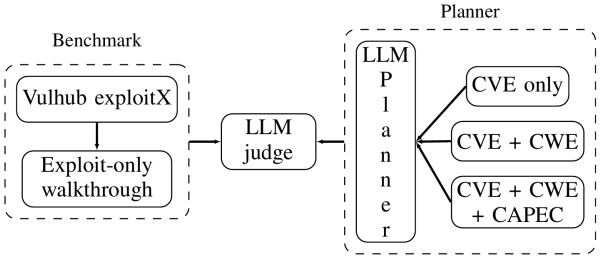

Figure 1. Motivation experiment pipeline. Vulhub exploit walkthroughs are normalized into an exploit-only gold target. The planner is evaluated under four input conditions, and a comparative LLM judge determines which condition best matches the gold target.

4.2. Experiment Structure

Recent agentic penetration testing frameworks increasingly rely on multi-stage reasoning pipelines in which planning is a critical component for successful exploitation [27]–[29]. In particular, planning modules are responsible for transforming vulnerability information into actionable exploitation procedures. However, existing approaches typically rely only on CVE and CWE information, despite CAPEC explicitly describing how weaknesses can be operationalized through adversarial attack patterns.

To evaluate whether CAPEC provides useful procedural knowledge for exploitation planning, we design a controlled experiment that isolates the contribution of CAPEC information during the planning stage. More specifically, we investigate whether augmenting vulnerability descriptions with CAPEC information improves the quality of LLM-generated exploitation plans. The experiment consists of three stages: benchmark construction, plan generation, and comparative evaluation.

4.2.1. Benchmark Construction. We construct the benchmark from manually curated exploitation walkthroughs derived from Vulhub [30]. We use Vulhub as a reproducible and standardized source of expert-grounded exploitation procedures. Each instance is associated with CVE, CWE, and CAPEC metadata and paired with an expert-written exploitation walkthrough. To avoid benchmark-specific artifacts, we derive the ground-truth target exclusively from the `exploit section` of each walkthrough, excluding setup instructions, container configuration, and environment-specific reproduction details. The resulting representation is converted into a structured comparison that represents the ground-truth, which is used for evaluation.

- **Overall**: overall alignment with the gold-standard exploitation plan;
- **Attack goal**: identification of the primary exploitation objective;
- **Entry point**: identification of the vulnerable interface, endpoint, parameter, or component;
- **Core actions**: inclusion of the essential exploitation steps;
- **Mitigations**: identification of relevant defensive countermeasures or remediation actions.

4.2.2. Plan Generation. For each benchmark instance, an LLM-based planner (inspired by [27]–[29]) generates a structured exploitation plan using only the information provided under a specific input condition. The planner never receives access to the original Vulhub walkthrough or the hidden gold-standard reference plan.

To explicitly isolate the effect of CAPEC knowledge, we restrict this experiment to vulnerabilities with existing ground-truth CWE–CAPEC mappings. Vulnerabilities lacking CAPEC associations are instead reserved for the downstream evaluation presented in Section 8.4, where we assess whether *predicted* CWE–CAPEC links improve exploitation planning. For each vulnerability, we generate plans under three increasingly informative conditions: **CVE**, using only CVE-level information; **CVE+CWE**, adding the associated weakness; and **CVE+CWE+CAPEC**, further adding attack-pattern information. The planner is tasked with

generating an exploitation walkthrough structured according to the same categories used in the ground-truth representation, enabling direct comparison during evaluation.

4.2.3. Evaluation. To evaluate generated plans, we adopt an LLM-as-a-judge protocol using comparative assessment against the reference plan described in Section4.2.1. Recent work has shown that strong LLM judges can achieve agreement rates with human evaluators comparable to inter-human agreement, particularly in pairwise settings [31]. Prior studies further show that comparative evaluation is substantially more reliable than absolute scoring for open-ended generation tasks, especially when structured evaluation criteria and reference targets are provided [31].

Our evaluation setting closely follows these validated conditions. The judge receives the same vulnerability instance, the hidden gold-standard exploitation plan, and the candidate plans generated under different information conditions. The judge then performs comparative selection according to explicitly defined planning criteria.

TABLE 1. COMPARISON OF SCENARIO WINNERS ACROSS 168 RECORDS. VALUES ARE REPORTED AS COUNT AND PERCENTAGE.

Field	CVE	CVE+CWE	CVE+CWE+ CAPEC
Overall	12 (7.1%)	45 (26.8%)	**111 (66.1%)**
Attack goal	16 (9.5%)	51 (30.4%)	**101 (60.1%)**
Entry point	18 (10.7%)	59 (35.1%)	**91 (54.2%)**
Core actions	15 (8.9%)	46 (27.4%)	**107 (63.7%)**
Mitigations	37 (22.0%)	37 (22.0%)	**88 (52.4%)**

Table 1 reports the results over 168 benchmark instances. Across all evaluation dimensions, the CAPEC-enhanced condition consistently achieves the highest win frequency. In particular, the `CVE+CWE+CAPEC` condition is selected as the best overall plan in 66.1% of cases, substantially outperforming both the CVE-only and CVE+CWE settings.

The largest improvements are observed for *attack goal* and *core actions*, suggesting that CAPEC contributes procedural exploitation knowledge that is not captured by vulnerability descriptions or weakness classes alone. More broadly, these results indicate that CAPEC helps bridge the gap between identifying a vulnerability and reasoning about how it can be exploited in practice.

Overall, this experiment provides empirical motivation for completing missing CWE–CAPEC links. Since CAPEC-enhanced inputs improve exploitation planning when ground-truth mappings are available, recovering missing CWE–CAPEC relationships may expand the availability of procedural exploitation knowledge for automated security reasoning systems.

5. Methodology

Motivated by the observation that CAPEC knowledge improves downstream exploitation planning, we now address the core research question of this work:

> How can missing CWE–CAPEC relationships be recovered by leveraging existing cybersecurity knowledge?

We model the recovery of unobserved CWE–CAPEC associations as relation-specific link prediction over a heterogeneous cybersecurity graph. During evaluation, observed CWE–CAPEC edges are masked and predicted using the remaining graph structure and node information, while non-target relations are retained as contextual evidence.

This setting presents two main challenges. First, the target CWE–CAPEC relation is sparse, and the original CWE–CAPEC graph provides limited evidence for inferring unobserved associations. Second, relation types are highly imbalanced, which can cause dense relations to dominate both neighborhood aggregation and context generation.

To address these challenges, we propose CC-GATNE, a cybersecurity-oriented extension of GATNE [1]. CC-GATNE enriches the graph with CVE and ATT&CK Technique entities, incorporates text-derived semantic node features, replaces uniform neighbor aggregation with an attention-based mechanism, and introduces layer-balanced random walks. Together, these components provide additional structural and semantic evidence while reducing the influence of noisy neighborhoods and dominant relation types. Figure 2 summarizes the proposed framework.

5.1. GATNE

GATNE [1] is designed for attributed multiplex graphs in which nodes may participate in multiple relation types. Rather than learning a single representation for each node, GATNE combines a shared base embedding with a relation-specific component.

For a node v and relation type r, GATNE computes the representation:

$$\mathbf{h}_v^{(r)} = \mathbf{b}_v + \mathbf{M}_r^T \left(\mathbf{U}_v \mathbf{a}_{v,r}\right)$$

where $\mathbf{b}_v$ is the base embedding of node v, $\mathbf{U}_v$ is a set of relation-aware latent embeddings, $\mathbf{a}_{v,r}$ is an attention vector that determines the importance of these latent components for relation r, and $\mathbf{M}_r$ is a relation-specific transformation matrix.

The resulting representation allows the same node to be modeled differently across relation contexts. For example, a CWE may participate in hierarchical CWE–CWE relations, vulnerability-level CVE–CWE mappings, and exploitation-oriented CWE–CAPEC relations. Relation-specific representations of two nodes can subsequently be used to estimate whether an edge of a given type should connect them.

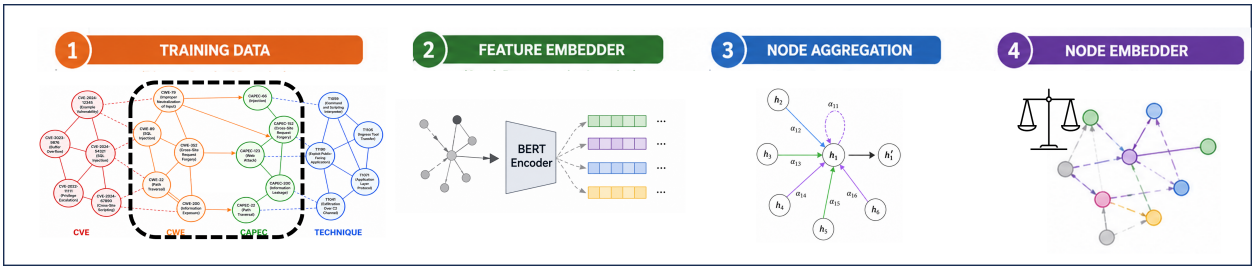

Figure 2. Overview of the proposed heterogeneous graph representation learning framework for cybersecurity knowledge integration. The proposed methodology enhances node representation learning through four complementary components: (1) the enrichment of the training graph with multi-domain cybersecurity entities and relations spanning CVEs, CWEs, CAPEC attack patterns, and adversarial techniques; (2) BERT-based semantic feature encoding to capture contextual information from textual node descriptions; (3) cross-graph attention-driven aggregation to align heterogeneous semantic and structural information into a unified embedding space; and (4) a layer-balanced random walk generation strategy that redistributes walks across heterogeneous edge types, mitigating relation imbalance and promoting more diverse and representative training samples.

5.2. CC-GATNE

CC-GATNE extends GATNE through four complementary components. First, graph augmentation introduces CVE and ATT&CK Technique entities and their associated relations. Second, semantic node initialization incorporates information from textual entity descriptions. Third, attention-based neighbor aggregation assigns different importance to sampled neighbors. Finally, layer-balanced random walks reduce the dominance of densely represented relation types during context generation.

The following subsections describe each component in detail.

5.2.1. Knowledge Augmentation. The standard CWE–CAPEC graph captures direct relationships between weakness classes (CWEs) and attack patterns (CAPECs), but offers limited support for identifying missing or indirect associations. Building on the frameworks proposed by [32], [33], which connect CVEs, CWEs, CAPECs, and tactics, techniques, and procedures (TTPs), we augment the graph with two complementary sources of cybersecurity knowledge: Common Vulnerabilities and Exposures (CVEs) and techniques from the MITRE ATT&CK framework.

CVEs provide concrete vulnerability instances that ground abstract CWE categories in real-world software flaws. This allows the model to distinguish between CWEs that may be textually similar but occur in different vulnerability contexts.

MITRE techniques provide adversarial behavior context by describing how attackers execute techniques during exploitation. By linking attack patterns to TTP-related knowledge, the graph gains additional behavioral signals that are closer to exploitation practice than to weakness descriptions alone.

Together, CVEs and MITRE TTPs create additional multi-hop paths between CWE and CAPEC nodes. For example, a CWE may be associated with a set of CVEs, which in turn expose recurring exploitation behaviors related to specific attack patterns or techniques. These indirect paths provide structural evidence that may reveal plausible CWE–CAPEC relationships even when no direct mapping is available.

5.2.2. Semantic Feature Enrichment. Although graph structure captures relational dependencies between cybersecurity entities, it does not fully exploit the semantic information contained in textual descriptions. CWE and CAPEC entries often encode detailed information about vulnerability characteristics, exploitation behavior, prerequisites, and mitigation strategies that may not be directly recoverable from graph connectivity alone. We provide node features with contextual text embeddings extracted from the corresponding CWE and CAPEC descriptions. Given the textual description x_v associated with node v, we compute a semantic embedding:

$$\mathbf{s}_v = \text{BERT}(x_v),$$

where $\mathbf{s}_v \in \mathbb{R}^d$ captures contextual semantic information derived from the pretrained BERT encoder.

These semantic embeddings are integrated with the structural representations learned through heterogeneous message passing.

5.2.3. Attention-Based Neighbor Aggregation. In the original GATNE formulation, the aggregation of neighboring nodes for a given edge type is performed through a simple summation over sampled neighbors. This approach assumes that all neighboring nodes from the same edge type contribute equally to the representation of the target node. However, neighbors often carry different levels of semantic relevance. Consequently, assigning uniform importance to all neighbors may dilute informative signals and amplify noisy or less relevant connections.

To address this limitation, we replace the uniform aggregation mechanism with an attention-based neighbor aggregation strategy. The proposed mechanism enables the model to learn adaptive importance weights

for each neighbor, allowing more informative neighbors to contribute more strongly to the final representation.

Let $x_{btn} \in \mathbb{R}^{d_u}$ denote the embedding of the n-th sampled neighbor of node b under edge type t, where d_u is the edge-specific embedding dimension. For each neighbor embedding, an attention score is computed using a two-layer neural attention mechanism:

$$e_{btn} = W_2\sigma(W_1 x_{btn} + b_1) + b_2$$

where:

- $W_1 \in \mathbb{R}^{d_a \times d_u}$ projects neighbor embeddings into an intermediate attention space of dimension d_a,
- $W_2 \in \mathbb{R}^{1 \times d_a}$ maps the transformed representation into a scalar compatibility score,
- b_1 and b_2 are learnable bias terms,
- $\sigma(\cdot)$ introduces non-linearity with the function $\tanh$.

The resulting attention score e_{btn} quantifies the importance of neighbor n for the representation of node b under relation type t. The scores are normalized across all sampled neighbors through a softmax operation, and the resulting attention weights are used to compute the final edge-type-specific representation as a weighted aggregation of neighboring embeddings.

Unlike mean or sum aggregation, this formulation enables the model to dynamically emphasize semantically important neighbors while suppressing noisy or weakly informative ones. This is particularly beneficial in heterogeneous graphs, where neighbors connected through the same edge type may still vary substantially in relevance.

5.2.4. Random Walk Balancing. Standard random walk strategies in heterogeneous graphs are often biased toward densely connected edge types, causing frequent relations to dominate context generation while underrepresenting sparse but semantically important relations. In cybersecurity knowledge graphs, this imbalance is particularly problematic due to the highly uneven distribution of relations across vulnerabilities, weaknesses, attack patterns, and techniques.

To mitigate this issue, we introduce a layer-balanced random walk strategy that redistributes transition probabilities across edge types during walk generation. Instead of sampling neighbors solely based on local connectivity, the proposed approach encourages a more uniform exploration of heterogeneous graph layers, ensuring that less frequent relations contribute more consistently to node contexts.

Formally, let $r \in \mathcal{R}$ denote an edge type and let $d_r(v)$ represent the number of neighbors of node v connected through relation r. During random walk generation, the transition probability from node u to node v through relation r is adjusted as:

$$P(v \mid u, r) \propto \frac{1}{d_r(u)^\alpha},$$

where α controls the balancing strength. Higher values of α reduce the dominance of highly connected relation types, promoting more diverse exploration across heterogeneous graph layers.

6. Experimental Setup

6.1. Dataset Overview

We construct a heterogeneous cybersecurity graph to support CWE–CAPEC link prediction. The graph combines weakness classes, attack patterns, vulnerability instances, and adversarial techniques from established cybersecurity knowledge bases. Specifically, it includes 956 CWE entries and 615 CAPEC entries derived from the official MITRE frameworks.

We further incorporate 19,264 CVE instances associated with CWE labels published in 2023. This design choice provides recent CVE–CWE mappings while avoiding the severe labeling sparsity observed in more recent years, where many CVEs remain unlabeled.

To enrich the graph with adversarial behavior knowledge, we also incorporate data from the MITRE ATT&CK framework, including 188 techniques. In Section 8, we additionally analyze the impact of integrating the 14 high-level ATT&CK tactics.

6.2. Graph Construction

The heterogeneous cybersecurity graph integrates relations across CVEs, CWEs, CAPECs, and ATT&CK techniques. We include both cross-framework relations (e.g., CVE–CWE, CWE–CAPEC, CAPEC–Technique) and intra-framework relations such as hierarchical dependencies within CWE, CAPEC, and ATT&CK ($\mathrm{CWE}_{\text{parent}}$-to-$\mathrm{CWE}_{\text{children}}$).

Although the graph contains multiple relation types, the prediction task focuses exclusively on missing CWE–CAPEC links. All remaining relations are retained during training as contextual structural information that supports heterogeneous representation learning and multi-hop reasoning across cybersecurity entities.

6.3. Evaluation setup

Since the CWE to CAPEC is a one-to-many link prediction problem, given that out of the CWEs that have a connection, roughly 50% of them are one-to-many connections, we create the dataset and evaluation accordingly to handle the problem adequately (48% of one-to-one, 52% of one-to-many). We formulate CWE–CAPEC link prediction as a binary link-prediction task on a graph. Each observed (CWE, CAPEC) association is encoded as a positive edge, while non-linked CWE–CAPEC pairs are used as negative instances during evaluation. Consequently, a CWE associated with multiple CAPECs appears in multiple

rows, one for each valid edge. This does not represent an artificial duplication of samples or a reformulation into multiple single-label problems; rather, it is the natural edge-level representation of a one-to-many relation in a link-prediction setting, where the model scores candidate node pairs independently. The train/test split is performed at the edge level, and held-out test edges are excluded from the graph structures used during training, including the random walks, to prevent information leakage. To obtain a more robust estimate of performance, we adopt a 5-fold cross-validation protocol in which only the target relation (CWE-CAPEC) is evaluated, while edges of other relation types are retained in training to preserve the heterogeneous graph context. Furthermore, each fold-based experiment is repeated 6 times with different random seeds, resulting in 30 runs per model/configuration. The final reported results are aggregated across these runs, thereby reducing sensitivity to a particular partition or stochastic training effect and enabling statistical comparison between models.

We use Area Under the Curve (AUC) as validation metric because it measures the model's ability to rank true associations above non-existing ones independently of a fixed decision threshold.

7. Results

In this section, we evaluate the effectiveness of the proposed CC-GATNE framework and analyze the contribution of each component introduced in Section 5. Our evaluation is structured progressively to isolate the impact of both data-level and model-level enhancements over the original GATNE architecture.

Table 2 summarizes the performance of the baseline GATNE model together with the successive improvements introduced in this work. We first establish the performance of the original GATNE configuration and then perform an ablation study to assess how each proposed component contributes to CWE–CAPEC link prediction performance.

We begin by evaluating the impact of the proposed data-level enhancements. In particular, we augment the heterogeneous cybersecurity graph with CVE and ATT&CK Technique nodes together with their associated relations (Section 5.2.1), while still training and evaluating exclusively on the CWE–CAPEC prediction task. We then investigate the effect of replacing one-hot node representations with contextual semantic embeddings generated using BERT (Section 5.2.2).

Building on the best-performing graph and feature configuration, we subsequently evaluate two model-level improvements: the attention-based aggregation mechanism (Section 5.2.3) and the proposed layer-balanced random walk strategy (Section 5.2.4). This progressive evaluation allows us to analyze the individual and combined contributions of structural enrichment, semantic feature augmentation, and heterogeneous representation-learning improvements.

By combining all proposed enhancements, we obtain **CC-GATNE**, which achieves a **22.15%** improvement in AUC compared to the original GATNE baseline.

Model	Data improv.		Model improv.		AUC (%)
	Add. train	Feat.	Node att.	Walk enc.	
GATNE	x	x	x	x	59.18
	✓	x	x	x	65.75
	x	✓	x	x	68.45
	✓	✓	x	x	69.68
	✓	✓	x	✓	70.59
	✓	✓	✓	x	71.73
CC-GATNE	✓	✓	✓	✓	**72.14**

TABLE 2. ABLATION STUDY EVALUATING THE CONTRIBUTION OF DATA- AND MODEL-LEVEL IMPROVEMENTS TO THE PERFORMANCE OF GATNE.

The ablation study further reveals that data-level enhancements contribute more substantially than architectural modifications alone. In particular, adding CVE and ATT&CK Technique nodes increases the baseline AUC from 59.18% to 69.68%, while semantic feature enrichment using BERT embeddings further improves representation quality. These results suggest that exposing the model to a richer heterogeneous cybersecurity context is critical for recovering missing exploitation relationships.

The model-level improvements provide additional gains on top of this enriched representation space. Attention-based aggregation improves relation-aware neighborhood modeling, while the layer-balanced random walk strategy helps generate more diverse heterogeneous node contexts. Although the individual improvements are incremental, their combination consistently produces the strongest overall performance.

Overall, the results highlight two key findings. First, enriching the cybersecurity graph with additional heterogeneous entities and semantic node representations provides the largest performance gains, suggesting that CWE–CAPEC prediction benefits strongly from cross-domain structural and contextual information. Second, the proposed model-level improvements further refine representation learning by improving relation-aware aggregation and mitigating edge-type imbalance during context generation.

By progressively combining these enhancements, CC-GATNE achieves a final AUC of **72.14%**, corresponding to a **22%** improvement over the original GATNE baseline.

7.1. Statistical significance

To assess the contribution of the proposed model-level enhancements, we performed an ablation study on top of the improved GATNE configuration that already includes additional training data and node features. In particular, we evaluated the impact of the proposed

walk encoding strategy and node attention mechanism independently. Statistical significance was assessed using the Wilcoxon signed-rank test over 30 independent runs by comparing each variant against the corresponding GATNE baseline with identical data enhancements. Results show that both model-level contributions yield statistically significant improvements. Specifically, the walk encoding strategy improves the ROC-AUC from 69.68% to 70.59% ($p = 0.015$), while the node attention mechanism further increases the ROC-AUC to 71.73% ($p < 0.001$).

7.2. SOTA Comparison

We now compare CC-GATNE against the existing CWE–CAPEC mapping approaches discussed in Section 3. Since prior work primarily reports threshold-dependent classification metrics such as precision, recall, and F1, we adapt our evaluation protocol to enable a fair comparison with these methods.

Unlike traditional classifiers, CC-GATNE produces similarity scores for candidate CWE–CAPEC pairs rather than binary predictions. Consequently, computing precision, recall, and F1 requires selecting an operating threshold to convert similarity scores into predicted links. To obtain a decision boundary, we calibrate the threshold on the validation set. More specifically, we compute the precision–recall curve over validation scores and select the threshold t^* that maximizes the validation F1 score. This threshold is then applied unchanged to the test set to generate binary predictions for precision, recall, and F1 computation. For consistency, we apply the same validation-based threshold selection procedure to all evaluated baselines.

TABLE 3. COMPARISON OF CWE–CAPEC MAPPING METHODS. PRECISION, RECALL, AND F1 ARE MICRO-AVERAGED OVER PREDICTED CWE–CAPEC LINKS. THE LLM USED FOR THIS EXPERIMENT IS GPT-4O-MINI.

Approach	Embedder	Prec	Rec	F1
[17] [18]	TF-IDF	15.60	20.85	17.01
[17] [18]	SBERT	22.38	24.64	23.00
Bonomi et. al [19]	Distilroberta	21.41	16.25	18.48
LLM CoT	-	19.51	24.26	21.63
LLM Few-shot	-	15.80	21.28	18.14
LLM Single-shot	-	18.28	24.59	20.96
CC-GATNE (ours)	Bert	59.82	83.65	69.61

Table 3 reports the comparison against prior semantic-similarity and LLM-based approaches. The results show that CC-GATNE substantially outperforms all evaluated baselines across precision, recall, and F1. In particular, our approach achieves an F1 score of **69.61**, compared with 23.00 for the strongest prior baseline. This large performance gap suggests that heterogeneous graph reasoning provides substantially richer predictive signals than approaches relying solely on textual similarity or direct LLM prompting.

More importantly, these findings reinforce a broader trend already observed in other cybersecurity knowledge-linking tasks. Prior work on CVE-to-CWE mapping [2] and CTI-sentence-to-TTP alignment [6] similarly showed that pure semantic similarity is often insufficient for accurately connecting cybersecurity concepts across different frameworks. In these settings, semantically related descriptions do not necessarily correspond to operationally or procedurally equivalent security concepts. Instead, effective mapping requires modeling contextual, relational, and structural dependencies between entities.

Our results further support this observation. CWE–CAPEC prediction cannot be reduced to matching semantically similar textual descriptions alone, since meaningful exploitation relationships frequently emerge through indirect structural associations involving vulnerabilities, attack patterns, and adversarial techniques. This explains why heterogeneous graph-based approaches such as CC-GATNE substantially outperform purely text-based retrieval and prompting methods.

More broadly, the results support the central hypothesis of this work: recovering missing CWE–CAPEC relationships requires modeling the structural and multi-relational organization of cybersecurity knowledge rather than relying exclusively on semantic similarity between isolated descriptions.

7.3. Use case: Mapping CTI to CAPEC

Although CWE and CAPEC are curated taxonomies that evolve relatively slowly, the security evidence that analysts need to interpret changes continuously. Incident reports, SOC alerts, threat-intelligence reports, and analyst notes often describe observed attacker behavior in natural language, without explicitly identifying the underlying software weakness. In such settings, mapping the evidence to CAPEC can provide attack-pattern context that supports incident response, threat analysis, event correlation, and downstream mitigation planning.

To demonstrate this use case, we evaluate whether CC-GATNE can be used beyond CWE nodes to map behavioral threat-intelligence evidence directly to CAPEC. We use two sentence-level threat-intelligence datasets, TRAM and AnnoCTR, in which each sentence is associated with one or more ATT&CK techniques. Since CAPEC provides partial mappings between CAPEC entries and ATT&CK techniques, we use these mappings to derive sentence-to-CAPEC pairs. This produces a dataset of (X) sentence-to-CAPEC pairs covering (Y) CAPEC classes.

Given a sentence (s), we encode its natural-language content and project it into the same representation space used by CC-GATNE. We then rank CAPEC entries according to their similarity to the sentence representation. This setting differs from the original CWE-to-CAPEC

task: the input is no longer a curated weakness node, but a previously unseen piece of behavioral evidence. As a result, the experiment tests whether the learned graph representation can support source-agnostic CAPEC retrieval.

This use case is particularly relevant for operational security workflows. A real-time incident responder or SOC system may observe events such as account enumeration, suspicious authentication attempts, command execution, privilege escalation, or data exfiltration. These observations may not map cleanly to CWE, because they describe attacker behavior rather than the root-cause weakness. Directly mapping such evidence to CAPEC therefore avoids forcing an intermediate CWE prediction and instead enriches the event with attack-pattern context.

8. Discussion

To better understand the performance gains achieved by CC-GATNE, this section analyzes the contribution of both the proposed data and model enhancements. We first analyze graph augmentation strategies on CWE–CAPEC link prediction performance and evaluate the interaction between graph structure and semantic node representations. Then, we investigate how different GNN architectures compare with GATNE. We then analyze the architectural modifications introduced in CC-GATNE and discuss how these changes improve heterogeneous representation learning within cybersecurity knowledge graphs. Finally, we examine the predicted CWE–CAPEC links and discuss their implications for automated cybersecurity reasoning.

Before discussing the contributions in detail, we define the *data configurations* used throughout this section to analyze the impact of graph augmentation. As illustrated in Figure 3, each configuration represents a different combination of node and edge types included during training:

- **CC**: CWE and CAPEC only (standard data).
- **CCC**: Extends CC by incorporating CVE nodes and CVE–CWE relations.
- **CCT**: Extends CC by incorporating ATT&CK Technique nodes and their associated relations.
- **CCCT**: Extends CC by incorporating CVE and ATT&CK Technique.
- **CCTT**: Extends CCT by incorporating ATT&CK Tactic nodes and their relations to Techniques.
- **CCCTT**: Extends CC by incorporating CVE, Technique, and Tactic nodes.

8.1. CC-GATNE Data-Level Improvements

As discussed in Section 7, CC-GATNE extends the original GATNE framework through both data-level and model-level enhancements. From the data perspective, we investigate two complementary factors: (i) graph augmentation through additional cybersecurity entities and relations, and (ii) semantic feature enrichment through different node embedding strategies.

8.1.1. Contribution of Graph Augmentation. To assess the contribution of graph augmentation while controlling for semantic feature initialization, we compare the graph configurations separately under one-hot, BERT, and SecureBERT node representations. The results are shown in Figure 3.

Progressively enriching the heterogeneous cybersecurity graph consistently improves link prediction performance across all embedding strategies. In particular, the transition from the essential *CC* configuration to the fully augmented CCCTT graph produces steady AUC improvements, especially for one-hot and SecureBERT representations. These findings suggest that additional cybersecurity entities and relations provide useful structural evidence for recovering missing CWE–CAPEC links. Graph augmentation also reduces the performance gap between embedding methods. While semantic feature initialization remains an important factor, richer heterogeneous connectivity partially compensates for weaker node representations by exposing additional multi-hop relationships between vulnerabilities, weaknesses, attack patterns, and adversarial behaviors. This observation supports that effective CWE–CAPEC prediction depends not only on textual similarity, but also on modeling the broader structural organization of cybersecurity knowledge.

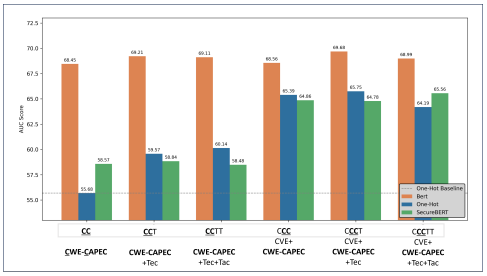

Figure 3. AUC achieved by GATNE across different graph configurations and node-feature representations. One-hot representations benefit most from richer graph configurations, whereas the performance of BERT and SecureBERT varies less across configurations. The dashed horizontal line indicates the baseline performance obtained with one-hot node representations and the non-augmented dataset.

8.1.2. Encoder Improvements. As already observed in Figure 3 (left), the node embedding strategy has the largest impact on overall performance. In particular, replacing the standard one-hot node representations with contextual semantic embeddings provides substantial improvements, highlighting the importance of semantic initialization for CWE–CAPEC link prediction.

Comparing the three embedding strategies (one-hot, BERT, and SecureBERT), a clear pattern emerges. BERT- and SecureBERT-based representations achieve

strong performance from the early stages of training, whereas one-hot encoded features require substantially more optimization steps to converge. This behavior is expected since one-hot vectors provide no semantic information about cybersecurity entities, forcing the model to learn node relationships purely from graph structure. On average, training with one-hot representations requires more than twice the number of training steps compared with SecureBERT-based initialization (53 vs. 25 epochs).

Graph augmentation also interacts differently with the various embedding strategies. One-hot representations benefit the most from additional graph context, since richer connectivity partially compensates for the absence of semantic node features. In contrast, BERT and SecureBERT already provide semantically meaningful initialization, making the relative contribution of additional graph structure less pronounced.

BERT-based embeddings. Among the evaluated node-feature initializations, BERT provides the strongest performance. Although SecureBERT is trained on cybersecurity text, it does not improve over general-purpose BERT in our setting. To further examine this result, we evaluate an unsupervised CWE–CAPEC retrieval task based only on embedding similarity. For each CWE, we rank candidate CAPEC entries according to their embedding similarity and measure whether a known CWE–CAPEC association appears among the top-ranked candidates. This experiment does not use graph structure or supervised link-prediction training, and is intended only to compare the semantic spaces induced by the two encoders.

Appendix 12 reports the resulting Hit@k scores. BERT outperforms SecureBERT across all evaluated ranking thresholds. Although the absolute scores remain low, confirming that text similarity alone is insufficient for CWE–CAPEC completion, the relative difference shows that BERT provides a better semantic initialization for the high-level framework descriptions used in this task.

8.2. GNN comparison

Before exploring the improvements brought by CC-GATNE, this section analyzes why GATNE is the most suitable baseline among GNN approaches. To this end, Table 4 compares the GNN architectures described in Section 2 by characterizing their main structural components, including support for heterogeneous nodes and edges, single final embedding representation, shared representation with relational conditioniing. For each GNN architecture, the reported performance corresponds to its best-performing configuration across all 6 data configurations described at the beginning of this Section (e.g., CWE-CAPEC, CVE-CWE-CAPEC, CWE-CAPEC-Technique, and their combinations) and across the three tested embedding strategies: One-hot, BERT, and SecureBERT. The AUC results reported are averaged over 30 runs and using coss-validation as specified in Section 7.

This comparison allows us to identify which architectural features (e.g., typed nodes, typed edges) contribute most to link prediction performance in the cybersecurity knowledge graph setting under the best data configurations for that specific model. The full comparison across architecture, data configurations, and encoders is presented in Table 8 in the appendix.

Model	Heterogeneity Modeled	Single emb final represent	Shared representation with relation conditioning	Best config	AUC (%)
GCN [21]	None	✓	✗	BERT + cctt	58.79
GAT [22]	None	✓	✗	BERT + cct	55.21
R-GCN [23]	Edges	✓	✗	BERT + cc	58.33
HetGNN [25]	Nodes & edges	✓	✗	one-hot + ccctt	57.05
Simple-HGN [26]	Nodes & edges	✓	✓	one-hot + ccct	61.10
HGT [24]	Nodes & edges	✓	✗	BERT + cct	52.82
GATNE [1]	Nodes & edges	✗	✓	BERT + ccct	69.68

TABLE 4.
COMPARISON OF GNN MODELS ACROSS GRAPH STRUCTURE AND FEATURE SUPPORT.

The architectures in Table 4 can be organized along two main axes: how heterogeneity is incorporated into message passing and whether relation-specific information is preserved in the final representation. Along the first axis, GCN and GAT constitute homogeneous baselines, since they use shared aggregation operators without explicitly distinguishing node or edge types. R-GCN introduces edge heterogeneity through relation-specific transformation matrices, while HetGNN additionally models node-type differences through type-specific feature and neighborhood encoders. HGT represents the strictest form of heterogeneous message passing among the considered models, as its attention and message operators are conditioned on the complete source-node-type, edge-type, and target-node-type meta-relation. Along the second axis, Simple-HGN and GATNE follow a different design principle based on a shared node-representation space combined with relation conditioning. Simple-HGN incorporates edge-type embeddings into a shared GAT backbone, but aggregates the resulting relation-conditioned messages into a single final embedding per node. GATNE instead combines a base representation shared across relations with relation-specific components and retains a distinct final embedding for each node–relation pair. This distinction is consistent with the results in Table 4: GATNE achieves the highest AUC of 69.68%, followed by Simple-HGN at 61.10%, suggesting that a shared node representation may be particularly effective when relation-specific information is preserved rather than collapsed into a single embedding. However, this remains an empirical association based on the architectural characteristics identified in our analysis. Because the reported scores correspond to different optimal combinations of node features and graph augmentations, the performance differences cannot be attributed to the architectures alone. More broadly, the results indicate that graph augmentation and semantically informed node representations are beneficial on average. The following

section examines these effects separately, focusing on the improvements associated with semantic embeddings and augmentation strategies.

8.2.1. Homogeneous vs. Heterogeneous GNNs. To isolate the effect of modeling graph heterogeneity, we compare homogeneous and heterogeneous graph configurations across the evaluated data settings. Figure 4 reports their relative performance by averaging the AUC across embedding methods and across models within the homogeneous and heterogeneous GNN categories. This aggregation provides an overall view of whether explicitly preserving node and edge types yields consistent improvements beyond those attributable to a particular embedding method or model architecture.

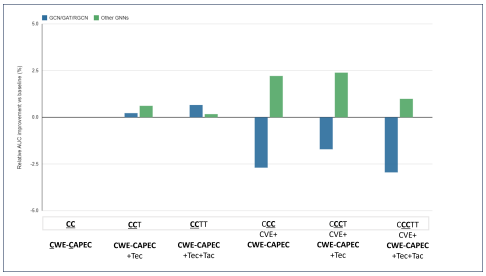

Figure 4. Relative performance changes of homogeneous and heterogeneous GNN architectures across the evaluated graph configurations, illustrating their sensitivity to the introduction of additional cybersecurity entities and relations.

Several important observations emerge from this analysis. First, the impact of graph augmentation strongly depends on both the type of cybersecurity knowledge introduced and the ability of the architecture to model heterogeneous relations. As shown in Figure 4, adding ATT&CK Techniques and Tactics generally improves performance across both homogeneous and heterogeneous GNNs, suggesting that these entities provide useful complementary exploitation context for CWE–CAPEC prediction.

However, the behavior changes substantially when CVE nodes are introduced. Since CVEs represent the largest entity type in the graph, architectures that cannot explicitly model node and edge heterogeneity (such as GCN, GAT, and, to a lesser extent, R-GCN) always experience performance degradation after CVE augmentation. Homogeneous GNNs aggregate neighborhood information without preserving relation awareness and, therefore, high-degree CVE neighborhoods can dominate message passing and introduce noisy or weakly relevant contextual signals. In contrast, heterogeneous architectures such as Simple-HGN and GATNE remain substantially more robust to CVE augmentation. By explicitly distinguishing node and edge types during representation learning, these models are better able to preserve heterogeneous relation semantics and selectively exploit the additional structural information introduced by large-scale vulnerability context.

8.2.2. Graph augmentation vs Encoder improvements. As observed in Section 8.1.1, the impact of semantic node initialization is substantial. As shown in Figure 5, where we averaged AUC over all the GNN architectures, using one-hot node representations generally benefits more from graph augmentation, since additional heterogeneous context partially compensates for the lack of semantic information in the initial node features. In contrast, semantically enriched encoders such as BERT and SecureBERT already provide informative contextual representations from the start, making the relative contribution of additional graph structure less pronounced.

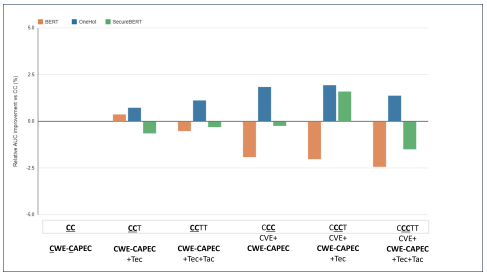

Figure 5. Average performance improvement over the base *CC* configuration across models, averaged and grouped by node-feature representation.

Overall, the results indicate that effective CWE–CAPEC prediction requires jointly modeling heterogeneous graph structure, relation-specific semantics, and contextual node representations.

8.3. CC-GATNE Model-level Improvements

The model-level enhancements provide smaller but consistent gains over the data-level improvements. Their main role is not to add new information to the graph, but to improve how GATNE uses the existing heterogeneous context. The two introduced improvements are: attention-based neighbor aggregation and layer-balanced random walks.

First, we replace GATNE's uniform neighbor aggregation with a learned attention mechanism. Uniform aggregation treats all sampled neighbors as equally informative, which can dilute useful signals in noisy heterogeneous neighborhoods. In contrast, attention allows the model to assign higher weight to neighbors that are more predictive for the target CWE–CAPEC relation. To examine this effect, we analyze flipped prediction cases, where the attention-based model predicts the correct link while mean aggregation fails. Figure 6 shows that attention concentrates most of the contribution mass on a small set of top-ranked neighbors and sharply reduces the influence of tail neighbors. The lower entropy and higher Gini coefficient confirm that the model learns a more selective neighborhood representation. This suggests that the gain comes from filtering the

sampled context, rather than from uniformly improving all neighbor contributions.

Second, we introduce layer-balanced random walks to reduce relation imbalance during context generation. In the original GATNE sampling process, dense edge types can dominate the generated walks, causing sparse but semantically important relations to be underrepresented. Our strategy redistributes part of the walk budget across relation layers, increasing the exposure of low-frequency edge types while reducing the dominance of the densest ones. Although the redistribution is moderate, with some sparse relations increasing by up to approximately 2%, it produces more diverse training contexts and better reflects the heterogeneous structure of the cybersecurity graph.

Overall, these changes make representation learning more selective and less biased toward ineffectual relations. While their impact is smaller than graph enrichment and semantic node initialization, they provide complementary gains by improving how the model exploits heterogeneous neighborhood information.

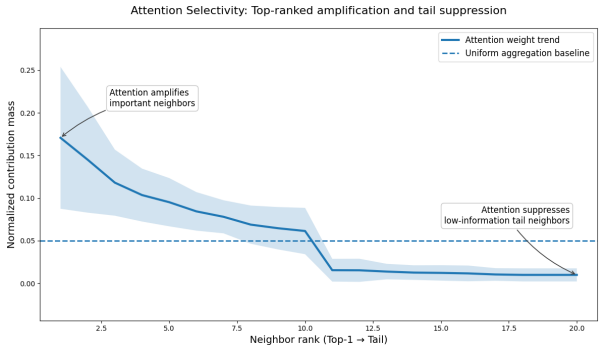

Figure 6. Attention selectivity on flipped prediction cases, where attention-based aggregation correctly predicts the target relation while mean aggregation fails. The attention mechanism concentrates contribution mass on top-ranked neighbors and suppresses tail neighbors, indicating that it improves representation quality by selectively amplifying informative context rather than averaging all sampled neighbors uniformly.

8.4. Downstream Example and Manual Validation

The motivation experiment in Section 4 showed that CAPEC information improves LLM-generated exploitation planning when ground-truth CWE–CAPEC mappings are available. We next examine whether *predicted* CAPEC associations inferred by CC-GATNE can provide useful exploitation context when no explicit CAPEC mapping exists.

We reuse the evaluation pipeline from Section 4.1 on the subset of Vulhub instances whose associated CWE has no existing CAPEC mapping in the original knowledge base. This subset contains 11 instances. For each instance, CC-GATNE ranks candidate CAPEC entries for the corresponding CWE. We restrict downstream to use up to the top-5 ranked predictions for each CWE.

In this experiment (which is consistent with the one in Section 4.1), the judge compares plans generated under three input conditions: CVE-only, CVE+CWE, and CVE+CWE+*Predicted CAPEC*. The results follow the same trend observed in the motivation experiment: adding predicted CAPEC information generally improves exploitation-plan quality compared with using only CVE or CVE+CWE information. However, the gains are smaller than those obtained with ground-truth CAPEC mappings. Detailed results are reported in Table 7.

To better understand this behavior, we complement the downstream experiment with an expert assessment of the predicted CWE–CAPEC associations. The annotation unit is the tuple consisting of the CVE, its associated CWE, the top-ranked predicted CAPEC, and the exploit-only Vulhub reference walkthrough. The evaluator assessed whether each predicted CAPEC was an exact match, semantically related, useful for exploitation planning, or potentially misleading, as fully defined in Appendix 12.1.

TABLE 5. EXPERT ASSESSMENT OF PREDICTED CWE–CAPEC RELATIONSHIPS FOR VULNERABILITIES WITHOUT EXISTING CAPEC MAPPINGS.

Criterion	Count	Percentage
Exact match	2/11	18.2%
Semantically related	1/11	9.1%
Useful for exploitation planning	6/11	54.5%
Potentially misleading	2/11	18.2%

The expert analysis shows that predicted CAPEC links should not be interpreted only through exact-match correctness. Several predictions that differ from the expected mapping still provide useful exploitation context, suggesting that CC-GATNE often recovers attack-pattern knowledge that is procedurally related to the target weakness. At the same time, the presence of potentially misleading predictions highlights the need for improvements in the mapping method and potentially in an enlargement of the CAPEC taxonomy.

9. Limitations

Despite the strong performance gains achieved by CC-GATNE, several limitations remain. First, the graph augmentation relies on only a subset of the broader cybersecurity and threat-intelligence sources that are potentially available. We restrict the analysis to frameworks that provide sufficiently reliable and structured relationships for integration [32], [33]. Consequently, the resulting graph does not capture the full diversity of cybersecurity entities, relations, and operational evidence. Extending the graph with additional threat-intelligence repositories, malware knowledge bases, and

real-world attack telemetry could provide richer contextual information and further improve representation quality.

Second, the downstream evaluation is limited by the relatively small number of vulnerabilities without existing CAPEC mappings available in the benchmark. While the qualitative and quantitative results suggest that predicted CAPEC links can improve exploitation planning, larger-scale expert evaluation is necessary to better assess the practical utility and reliability of inferred attack-pattern associations.

Finally, the proposed framework primarily models static structural relationships between cybersecurity entities. However, cybersecurity knowledge evolves continuously over time, with new vulnerabilities, attack techniques, and exploitation procedures emerging regularly. Incorporating temporal and dynamic graph modeling represents an important direction for future work.

10. Conclusions

This paper presented CC-GATNE, a heterogeneous graph-learning framework for predicting missing CWE–CAPEC relationships. We modeled cybersecurity knowledge as a multi-relational graph integrating vulnerabilities, weaknesses, attack patterns, and adversarial techniques, and extended GATNE through both data-level and model-level enhancements, including semantic node enrichment, attention-based aggregation, and layer-balanced random walks.

Our results show that heterogeneous graph reasoning substantially outperforms semantic-similarity and LLM-based approaches for CWE–CAPEC mapping. In particular, the experiments demonstrate that effective prediction requires jointly modeling structural dependencies, heterogeneous relation semantics, and contextual cybersecurity knowledge rather than relying solely on textual similarity.

Beyond link prediction accuracy, we further showed that CAPEC information provides measurable downstream benefits for LLM-based exploitation planning. Importantly, even predicted CAPEC associations inferred by CC-GATNE can improve procedural reasoning, suggesting that recovering missing cybersecurity relationships has practical value for automated security analysis and agentic penetration testing systems.

Overall, the findings of this work highlight the importance of heterogeneous graph reasoning for cybersecurity knowledge integration and suggest that completing missing links across security frameworks can directly support more capable automated cybersecurity reasoning systems.

References

[1] Y. Cen, X. Zou, J. Zhang, H. Yang, J. Zhou, and J. Tang, "Representation learning for attributed multiplex heterogeneous network," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery amp; Data Mining*, ser. KDD '19. ACM, Jul. 2019, p. 1358–1368. [Online]. Available: http://dx.doi.org/10.1145/3292500.3330964

[2] S. Simonetto, T. S. van Ede, P. Bosch, and W. Jonker, "Text2weak: mapping cves to cwes using description embeddings analysis," in *4th Workshop on Artificial Intelligence-Enabled Cybersecurity Analytics*, 2024.

[3] S. Simonetto, R. Oostveen, T. van Ede, P. Bosch, and W. Jonker, "Knowing your weaknesses is your greatest strength: Mapping cve to cwe by leveraging cwe hierarchy and llms," in *21st ACM ASIA Conference on Computer and Communications Security, ACM ASIACCS 2026*. ACM Press, 2026.

[4] H. Turtiainen and A. Costin, "Vulnberta: On automating cwe weakness assignment and improving the quality of cybersecurity cve vulnerabilities through ml/nlp," in *2024 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW)*. IEEE, 2024, pp. 618–625.

[5] E. Aghaei, W. Shadid, and E. Al-Shaer, "Threatzoom: Hierarchical neural network for cves to cwes classification," in *International Conference on Security and Privacy in Communication Systems*. Springer, 2020, pp. 23–41.

[6] M. Büchel, T. Paladini, S. Longari, M. Carminati, S. Zanero, H. Binyamini, G. Engelberg, D. Klein, G. Guizzardi, M. Caselli *et al.*, "{SoK}: Automated {TTP} extraction from {CTI} reports–are we there yet?" in *34th USENIX Security Symposium (USENIX Security 25)*, 2025, pp. 4621–4641.

[7] Z. Li, J. Zeng, Y. Chen, and Z. Liang, "Attackg: Constructing technique knowledge graph from cyber threat intelligence reports," in *European Symposium on Research in Computer Security*. Springer, 2022, pp. 589–609.

[8] G. Husari, E. Al-Shaer, M. Ahmed, B. Chu, and X. Niu, "Ttpdrill: Automatic and accurate extraction of threat actions from unstructured text of cti sources," in *Proceedings of the 33rd annual computer security applications conference*, 2017, pp. 103–115.

[9] P. Gao, X. Liu, E. Choi, S. Ma, X. Yang, Z. Ji, Z. Zhang, and D. Song, "Threatkg: A threat knowledge graph for automated open-source cyber threat intelligence gathering and management," *arXiv preprint arXiv:2212.10388*, 2022.

[10] V. Legoy, M. Caselli, C. Seifert, and A. Peter, "Automated retrieval of att&ck tactics and techniques for cyber threat reports," *arXiv preprint arXiv:2004.14322*, 2020.

[11] N. Rani, B. Saha, V. Maurya, and S. K. Shukla, "Ttphunter: Automated extraction of actionable intelligence as ttps from narrative threat reports," in *Proceedings of the 2023 australasian computer science week*, 2023, pp. 126–134.

[12] Y. You, J. Jiang, Z. Jiang, P. Yang, B. Liu, H. Feng, X. Wang, and N. Li, "Tim: threat context-enhanced ttp intelligence mining on unstructured threat data," *Cybersecurity*, vol. 5, no. 1, p. 3, 2022.

[13] F. I. Rahman, S. M. Halim, A. Singhal, and L. Khan, "Alert: A framework for efficient extraction of attack techniques from cyber threat intelligence reports using active learning," in *IFIP Annual Conference on Data and Applications Security and Privacy*. Springer, 2024, pp. 203–220.

[14] Z. Hao, C. Li, X. Fu, B. Luo, and X. Du, "Leveraging hierarchies: Hmcat for efficiently mapping cti to attack techniques," in *European Symposium on Research in Computer Security*. Springer, 2024, pp. 65–85.

[15] U. Kumarasinghe, A. Lekssays, H. T. Sencar, S. Boughorbel, C. Elvitigala, and P. Nakov, "Semantic ranking for automated adversarial technique annotation in security text," in *Proceedings of the 19th ACM Asia Conference on Computer and Communications Security*, 2024, pp. 49–62.

[16] B. Andrei, "Semantic model of attacks and vulnerabilities based on capec and cwe dictionaries," *International Journal of Open Information Technologies*, vol. 7, no. 3, pp. 38–41, 2019.

[17] K. Kanakogi, H. Washizaki, Y. Fukazawa, S. Ogata, T. Okubo, T. Kato, H. Kanuka, A. Hazeyama, and N. Yoshioka, "Tracing cve vulnerability information to capec attack patterns using natural language processing techniques," *Information*, vol. 12, no. 8, p. 298, 2021.

[18] ——, "Comparative evaluation of nlp-based approaches for linking capec attack patterns from cve vulnerability information," *Applied Sciences*, vol. 12, no. 7, p. 3400, 2022.

[19] S. Bonomi, A. Ciavotta, S. Lenti, and A. Palma, "Beyond the surface: An nlp-based methodology to automatically estimate cve relevance for capec attack patterns," *arXiv e-prints*, pp. arXiv–2501, 2025.

[20] W. Ren, Y. Lei, Y. Hong, X. Song, J. Yao, Y. Du, and W. Li, "Network attack knowledge reasoning model based on graph convolutional neural networks," 2023.

[21] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," 2017. [Online]. Available: https://arxiv.org/abs/1609.02907

[22] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph attention networks," 2018. [Online]. Available: https://arxiv.org/abs/1710.10903

[23] M. Schlichtkrull, T. N. Kipf, P. Bloem, R. van den Berg, I. Titov, and M. Welling, "Modeling relational data with graph convolutional networks," 2017. [Online]. Available: https://arxiv.org/abs/1703.06103

[24] Z. Hu, Y. Dong, K. Wang, and Y. Sun, "Heterogeneous graph transformer," 2020. [Online]. Available: https://arxiv.org/abs/2003.01332

[25] C. Zhang, D. Song, C. Huang, A. Swami, and N. V. Chawla, "Heterogeneous graph neural network," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, ser. KDD '19. New York, NY, USA: Association for Computing Machinery, 2019, p. 793–803. [Online]. Available: https://doi.org/10.1145/3292500.3330961

[26] Q. Lv, M. Ding, Q. Liu, Y. Chen, W. Feng, S. He, C. Zhou, J. Jiang, Y. Dong, and J. Tang, "Are we really making much progress? revisiting, benchmarking and refining heterogeneous graph neural networks," in *Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining*, 2021, pp. 1150–1160.

[27] X. Shen, L. Wang, Z. Li, Y. Chen, W. Zhao, D. Sun, J. Wang, and W. Ruan, "Pentestagent: Incorporating llm agents to automated penetration testing," in *Proceedings of the 20th ACM Asia Conference on Computer and Communications Security*, 2025, pp. 375–391.

[28] G. Deng, Y. Liu, V. Mayoral-Vilches, P. Liu, Y. Li, Y. Xu, T. Zhang, Y. Liu, M. Pinzger, and S. Rass, "{PentestGPT}: Evaluating and harnessing large language models for automated penetration testing," in *33rd USENIX Security Symposium (USENIX Security 24)*, 2024, pp. 847–864.

[29] Y. Ginige, A. Niroshan, S. Jain, and S. Seneviratne, "Autopentester: An llm agent-based framework for automated pentesting," in *2025 IEEE 24th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*. IEEE, 2025, pp. 163–174.

[30] "Vulhub," 2026, pre-built vulnerable environments based on Docker-Compose. [Online]. Available: https://github.com/vulhub/vulhub

[31] L. Zheng, W.-L. Chiang, Y. Sheng, S. Zhuang, Z. Wu, Y. Zhuang, Z. Lin, Z. Li, D. Li, E. Xing *et al.*, "Judging llm-as-a-judge with mt-bench and chatbot arena," *Advances in neural information processing systems*, vol. 36, pp. 46 595–46 623, 2023.

[32] Anonymous, "CVE to MITRE Dataset," https://drive.google.com/drive/folders/1Q1lA8Gh6MwbYsyUjhJi7WL_e-gMUtIeD?usp=sharing, 2024.

[33] E. Hemberg, J. Kelly, M. Shlapentokh-Rothman, B. Reinstadler, K. Xu, N. Rutar, and U.-M. O'Reilly, "Linking threat tactics, techniques, and patterns with defensive weaknesses, vulnerabilities and affected platform configurations for cyber hunting," *arXiv preprint arXiv:2010.00533*, 2020.

11. Appendix

11.1. Graph Neural Networks

Graph Neural Networks (GNNs) are neural architectures designed to learn low-dimensional vector representations of graph entities by jointly exploiting node features and graph topology. Unlike traditional machine learning methods that operate on isolated instances, GNNs propagate information across graph neighborhoods, enabling nodes to incorporate structural and semantic context from connected entities.

Formally, let $G = (V, E)$ denote a graph where V is the set of nodes and E is the set of edges. Each node $v \in V$ is associated with an initial feature representation $\mathbf{x}_v$. A GNN iteratively updates node embeddings through neighborhood aggregation, allowing each node representation to encode information from its local graph structure.

At layer $l+1$, a generic message-passing operation can be expressed as:

$$\mathbf{h}_i^{(l+1)} = \sigma\left(\text{AGG}\left(\left\{\mathbf{W}^{(l)}\mathbf{h}_j^{(l)} : j \in \mathcal{N}(i)\right\}\right)\right),$$

where $\mathcal{N}(i)$ denotes the neighborhood of node i, $\mathbf{W}^{(l)}$ is a trainable transformation matrix, $\text{AGG}(\cdot)$ represents a permutation-invariant aggregation function such as mean, sum, or attention-based aggregation, and $\sigma(\cdot)$ is a non-linear activation function. By stacking multiple layers, GNNs enable nodes to progressively capture information from multi-hop neighborhoods.

While early GNN architectures were primarily designed for homogeneous graphs containing a single node and edge type, cybersecurity knowledge graphs are inherently heterogeneous. Entities such as vulnerabilities, weaknesses, attack patterns, techniques, tactics, and software products exhibit distinct semantic roles and are connected through multiple relation types. Modeling all entities and relations identically may therefore obscure important structural dependencies and relation semantics.

To address this limitation, heterogeneous GNNs extend message passing to multi-relational settings by incorporating node-type and edge-type awareness during aggregation and representation learning. This enables the model to learn different propagation patterns depending on the semantic nature of the entities and relations involved.

We model cybersecurity knowledge as a heterogeneous graph

$$G = (V, E, \phi, \psi),$$

where V denotes the set of nodes, E denotes the set of edges, $\phi : V \to \mathcal{A}$ maps each node to a node type in the set of entity types $\mathcal{A}$, and $\psi : E \to \mathcal{R}$ maps each edge to a relation type in the set of relations $\mathcal{R}$.

In our setting, node types include cybersecurity entities such as:

$$\mathcal{A} = \{\texttt{CWE}, \texttt{CAPEC}, \texttt{CVE}, \texttt{Technique}, \dots\},$$

while edge types represent semantic relationships between these entities, such as:

$$\mathcal{R} = \{\texttt{exploits}, \texttt{related\_to}, \texttt{mapped\_to}, \dots\}.$$

Each node $v \in V$ is associated with an embedding vector

$$\mathbf{h}_v \in \mathbb{R}^d,$$

learned through heterogeneous message passing across the graph.

The objective of this work is to predict missing links between CWE and CAPEC nodes. Formally, given a partially observed heterogeneous graph G, we aim to learn a function

$$f : (u, v) \to [0, 1],$$

that estimates the likelihood that an edge exists between a CWE node u and a CAPEC node v.

After learning node embeddings, link prediction is performed by applying a scoring function over pairs of node representations. Given embeddings $\mathbf{h}_u$ and $\mathbf{h}_v$, the probability of a link can be estimated as:

$$\hat{y}_{uv} = \sigma\left(\text{score}(\mathbf{h}_u, \mathbf{h}_v)\right),$$

where score$(\cdot)$ corresponds to dot product and $\sigma(\cdot)$ maps the score into a probability value.

This formulation allows the model to infer previously unseen CWE–CAPEC relationships by leveraging both semantic node representations and the structural dependencies encoded within the heterogeneous cybersecurity graph.

11.2. Other Graph Neural Network Baselines

In addition to GATNE [1], we compare against representative graph neural network architectures for homogeneous, relational, and heterogeneous graph representation learning. These baselines differ in how they aggregate neighborhood information and model heterogeneous relations and node types.

Homogeneous GNNs. Graph Convolutional Networks (GCNs) [21] learn node embeddings by recursively aggregating features from neighboring nodes through a normalized adjacency matrix. By stacking multiple layers, GCNs capture information from multi-hop neighborhoods. However, standard GCNs assume homogeneous graphs and do not explicitly model node or edge heterogeneity.

Graph Attention Networks (GATs) [22] extend GCNs by introducing attention mechanisms over neighboring nodes, allowing the model to assign different importance weights during aggregation. Similar to GCNs, standard GATs are primarily designed for homogeneous graph settings.

Relational and Heterogeneous GNNs.. Relational GCNs (R-GCNs) [23] extend GCNs to multi-relational graphs through relation-specific transformation matrices, enabling relation-aware message passing across different edge types. Heterogeneous Graph Transformers (HGTs) [24] use type-specific projections and relation-specific attention mechanisms to model interactions across multiple node and edge types. HetGNN [25] captures heterogeneity by aggregating neighbors separately according to node types and combining the resulting type-specific representations into a unified embedding. Simple-HGN [26] extends graph attention to heterogeneous graphs by incorporating edge-type embeddings directly into the attention mechanism, providing a lightweight approach for heterogeneous representation learning.

Compared with these approaches, GATNE is particularly well suited for cybersecurity knowledge graphs because it explicitly models multiple relation types through edge-type-specific embeddings while simultaneously learning a shared representation across heterogeneous graph layers. Unlike homogeneous GNNs, GATNE preserves relation semantics during representation learning; unlike R-GCN and CompGCN, it is designed to operate efficiently on multiplex heterogeneous networks with distinct structural contexts across edge types. Furthermore, compared with transformer-based heterogeneous architectures such as HGT, GATNE provides a lightweight and scalable framework that naturally integrates heterogeneous random walks for context generation. These characteristics make GATNE especially suitable for modeling sparse and highly multi-relational cybersecurity graphs, where exploitation semantics often emerge through diverse cross-layer interactions rather than isolated pairwise relations.

11.3. Example: Semantic Alignment in CWE–CAPEC Mapping

A representative example is the mapping between CWE-120 (*Buffer Copy without Checking Size of Input*) and CAPEC-24 (*Filter Failure through Buffer Overflow*).

Qualitative analysis shows that BERT assigns stronger attention to semantically discriminative terms such as *buffer*, *copy*, and *checking*, which directly characterize the underlying weakness mechanism. In contrast, SecureBERT places relatively greater emphasis on broader security-related terminology such as *overflow*. Similarly, for the CAPEC description, BERT focuses more strongly on attack-specific behavioral concepts such as *filter*, *failure*, and *buffer*, whereas SecureBERT distributes attention more uniformly across generic security terminology.

These observations suggest that BERT captures more discriminative semantic relationships between CWE and CAPEC descriptions, while SecureBERT may overemphasize domain-associated cybersecurity terminology that is less informative for fine-grained weakness–attack-pattern mapping.

12. Encoder Analysis

In the main evaluation, general-purpose BERT consistently outperformed SecureBERT as a node-feature initializer. This result is noteworthy because SecureBERT is trained on cybersecurity corpora and might therefore be expected to better represent cybersecurity concepts. A plausible explanation is that the textual descriptions used in our graph correspond primarily to high-level security concepts, such as CWE weaknesses, CAPEC attack patterns, and ATTCK techniques, rather than low-level threat-intelligence artifacts such as indicators of compromise, malware reports, operational telemetry, or incident-specific descriptions. In this setting, broad semantic language understanding may transfer more effectively than representations specialized for lower-level cybersecurity text.

To further examine this behavior, we evaluate an unsupervised CWE–CAPEC retrieval task using only embedding similarity. For each CWE, we rank candidate CAPEC entries according to their embedding similarity and measure whether a known CWE–CAPEC association appears among the top-ranked candidates. This experiment does not use graph structure or supervised link-prediction training; it is intended only to compare the semantic spaces induced by the two encoders.

Table 6 reports the resulting Hit@k scores. BERT outperforms SecureBERT across all three ranking thresholds. Although the absolute scores are low, which confirms that text similarity alone is insufficient for CWE–CAPEC completion, the relative difference suggests that BERT better captures the high-level semantic relationships present in framework descriptions.

These results also clarify the role of semantic initialization in the full model. The encoder is not sufficient by itself to recover CWE–CAPEC links reliably, but it provides useful initial node representations that can be refined through heterogeneous graph structure. The improvement of BERT over SecureBERT therefore does not imply that general-purpose encoders are universally better for cybersecurity tasks. Rather, it suggests that encoder choice should match the linguistic level of the input data: high-level taxonomy descriptions may benefit more from general semantic representations, whereas lower-level CTI or telemetry data may require domain-specialized encoders.

We also inspected cases where BERT ranked the correct CAPEC substantially higher than SecureBERT. These cases typically involve broad conceptual similarity between weakness descriptions and attack-pattern descriptions, rather than domain-specific terminology. This supports the interpretation that general semantic coverage is particularly useful for the CWE–CAPEC setting.

TABLE 6. RANKING PERFORMANCE COMPARISON FOR UNSUPERVISED CWE–CAPEC SIMILARITY MATCHING.

Model	Hit@1	Hit@5	Hit@10
BERT	0.057	0.169	0.236
SecureBERT	0.036	0.134	0.203

TABLE 7. COMPARISON OF SCENARIO WINNERS ACROSS 11 RECORDS. VALUES ARE REPORTED AS COUNT AND PERCENTAGE.

Field	CVE	CVE+CWE	CVE+CWE+ CAPEC
Overall	1 (9.1%)	4 (36.4%)	**6 (54.5%)**
Attack goal	1 (9.1%)	4 (36.4%)	**6 (54.5%)**
Entry point	1 (9.1%)	**5 (45.5%)**	**5 (45.5%)**
Core actions	2 (18.2%)	4 (36.4%)	**5 (45.5%)**
Mitigations	**7 (63.6%)**	2 (18.2%)	1 (9.1%)

12.1. Expert evaluation criteria

We define these criteria as follows. *Exact Match* indicates that the predicted CAPEC corresponds to the attack pattern expected for the vulnerability. *Semantic Relatedness* indicates that the prediction captures a technically related exploitation mechanism, prerequisite, or attacker behavior even if it is not the exact expected pattern. *Exploitation Utility* indicates that the prediction provides information useful for identifying the attack goal, entry point, or core exploitation actions. *Potential Misleading Effect* indicates that the prediction suggests an irrelevant or inconsistent exploitation path that could degrade downstream reasoning.

12.2. Comparison across GNN architectures, embedding strategies, and configurations.

TABLE 8. AUC COMPARISON ACROSS GNN ARCHITECTURES, EMBEDDING STRATEGIES, AND HETEROGENEOUS GRAPH CONFIGURATIONS. RESULTS ARE REPORTED AS MEAN ± STANDARD DEVIATION ACROSS RUNS. THE UNDERLINED RESULT IS THE APPROXIMATED REPRODUCTION OF THE WORK FROM [20]

Model	Embedding	CC	CCC	CCT	CCCT	CCTT	CCCTT
GCN	One-hot	55.68±2.49	52.79±2.36	56.05±1.97	53.34±1.98	**56.64±3.42**	53.40±2.58
	BERT	57.69±2.10	55.72±2.26	58.43±2.43	56.17±2.34	**58.79±2.15**	56.48±3.33
	SecureBERT	53.03±1.78	52.04±3.25	54.18±2.14	53.24±1.76	**54.34±2.49**	51.37±2.29
GAT	One-hot	54.60±2.17	53.92±1.95	**55.19±2.07**	54.13±0.93	54.86±2.49	53.56±2.31
	BERT	54.39±2.74	51.43±1.44	**55.21±2.30**	52.27±2.21	54.91±1.74	52.42±3.04
	SecureBERT	52.46±2.05	51.05±2.09	51.76±1.08	**52.71±1.65**	53.00±1.26	50.39±1.97
R-GCN	One-hot	57.55±2.22	57.42±1.62	57.50±2.01	57.06±1.99	**58.00±4.26**	56.57±2.04
	BERT	**58.71±2.38**	58.10±2.83	58.14±2.13	57.54±3.11	57.08±2.51	57.58±3.60
	SecureBERT	**55.96±2.07**	54.22±2.09	54.64±1.74	55.06±1.36	55.67±1.99	53.60±1.61
HetGNN	One-hot	–	–	55.05±1.64	55.85±1.72	55.55±1.78	**57.05±2.93**
	BERT	–	–	**53.95±2.94**	54.71±0.63	53.46±3.16	52.14±2.88
	SecureBERT	–	–	54.62±1.92	**55.32±0.85**	52.86±3.30	51.98±2.69
HGT	One-hot	47.40±2.74	48.41±2.37	47.50±1.89	48.35±2.16	47.86±3.68	47.99±2.68
	BERT	50.68±2.32	51.78±2.70	**52.82±2.65**	51.24±2.46	52.18±2.50	52.57±2.73
	SecureBERT	48.42±2.44	48.33±2.62	47.93±2.91	**48.99±3.33**	48.38±1.81	47.27±2.41
Simple-HGN	One-hot	58.23±4.21	60.82±3.77	60.02±3.91	**61.10±3.47**	59.36±4.76	60.64±3.83
	BERT	54.94±4.22	52.80±1.95	**56.24±4.35**	52.74±2.01	54.90±4.17	52.53±3.03
	SecureBERT	**54.67±3.76**	51.81±2.62	52.54±2.52	52.85±1.76	53.05±1.83	51.14±1.54
GATNE	One-hot	59.18±8.07	65.39±3.69	59.57±8.11	**65.75±1.96**	60.14±7.93	64.19±3.65
	BERT	68.45±1.72	68.56±1.78	69.21±2.63	**69.68±2.21**	69.11±1.83	68.99±2.59
	SecureBERT	58.57±8.14	**64.86±2.63**	58.84±7.78	64.78±1.75	58.48±7.83	65.56±1.65