Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

Text-Aware Contrastive Learning for Bridging Graph Components in a Joint Embedding Space

Ahnaf Farhan

rownak.utep@gmail.com

The University of Texas at El Paso

Moqsadur Rahman

The University of Texas at El Paso

Monika Akbar

The University of Texas at El Paso

M. Shahriar Hossain

The University of Texas at El Paso

Research Article

Keywords: Graph Neural Network, Graph Contrastive Learning, Cybersecurity

Posted Date: March 26th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-8896676/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: No competing interests reported.

Text-Aware Contrastive Learning for Bridging Graph Components in a Joint Embedding Space

Ahnaf Farhan[1], Moqsadur Rahman[1], Monika Akbar[1], M. Shahriar Hossain[1]

[1*]Department of Computer Science, The University of Texas at El Paso, 500 W University Ave, El Paso, 79968, Texas, United States.

Contributing authors: rownak.utep@gmail.com; mrahman20@miners.utep.edu; makbar@utep.edu; mhossain@utep.edu;

Abstract

Graph-structured data across many domains consists of multiple heterogeneous components, each with its own features, that are related to one another even in the absence of explicit edges. In cybersecurity, for example, MITRE ATT&CK and CWE provide complementary graph-structured knowledge of adversarial behaviors and system weaknesses, each with rich text-annotated nodes but no explicit links between the graphs. This disconnect prevents learning joint embeddings and limits downstream tasks such as similarity search, retrieval, and automated reasoning. We propose AttWeakBridge, a text-aware Graph Contrastive Learning (GCL) framework for embedding disconnected, text-rich graph components into a shared semantic space. Unlike conventional GCL methods that operate on augmented views of a single connected graph, AttWeakBridge performs contrastive learning across separate graph components, combining node text embeddings with structural signals via a dual-encoder GNN and an inter- / intra-graph triplet sampling strategy. Experimental results on real-world cybersecurity graphs (MITRE ATT&CK and CWE) show that AttWeakBridge (i) improves similarity learning and cross-graph retrieval compared to text-only and structure-only baselines, and (ii) yields more coherent neighborhoods that align attack techniques with relevant weaknesses. This work contributes a general paradigm for embedding heterogeneous, disconnected graphs into a common semantic space.

Keywords: Graph Neural Network, Graph Contrastive Learning, Cybersecurity

1 Introduction

Modern data mining applications increasingly rely on graph-structured data to model complex relationships between entities. In many real-world settings, however, knowledge is fragmented in multiple graphs that describe different but related aspects of the same domain. These graph components are often disconnected, yet share rich textual metadata that implicitly encodes semantic relationships [1–3]. Learning meaningful representations that unify such disconnected, text-rich graphs remains a fundamental challenge in graph representation learning and knowledge discovery.

A motivating class of problems arises in domains where entities are described both by structured relationships and unstructured text, but lack explicit links across graph components [4, 5]. For example, in cybersecurity knowledge bases maintained by organizations such as MITRE, attack patterns and system weaknesses are curated as separate graph components—each internally structured and text-annotated, yet disconnected from one another. More generally, this setting exemplifies a broader data mining challenge: how to align multiple heterogeneous graph components using shared semantic signals without introducing artificial edges.

From a data mining perspective, the core challenge is to bridge multiple graph components into a shared embedding space while preserving both intra-graph structure and inter-graph semantic alignment. Traditional graph neural networks (GNNs) [6, 7] are not suitable for this task, as they rely on connectivity for message passing and fail when graphs are disconnected. Conversely, purely text-based methods overlook graph topology, leading to representations that are semantically plausible but structurally inconsistent.

Prior work on aligning heterogeneous or disconnected graphs typically relies on supervised link prediction, manual annotation, or graph augmentation strategies that introduce synthetic edges [8]. These approaches either lack scalability, depend on labeled data, or risk distorting the original graph structure. Hybrid methods that combine text and structure show promise, yet designing principled learning objectives that balance these two modalities—especially in the absence of inter-graph edges—remains an open research problem.

In this paper, we propose AttWeakBridge, a text-inspired Graph Contrastive Learning (GCL) framework that learns unified embeddings for disconnected graph components with text-rich nodes. While our experiments focus on two real-world cybersecurity graphs—MITRE ATT&CK and CWE—the proposed methodology is domain-agnostic and applicable to a broad range of data mining tasks involving fragmented graph data. AttWeakBridge leverages contrastive learning across graph components, using textual similarity to guide inter-graph alignment and structural similarity to preserve intra-graph coherence, without introducing explicit cross-graph edges.

We evaluate AttWeakBridge through experiments that examine (i) the quality of the learned embedding space, (ii) the model's ability to align semantically related nodes across graph components, and (iii) the robustness of the approach under different encoder and feature configurations.

In addition, we demonstrate how unified graph embeddings can support downstream retrieval and reasoning tasks through a lightweight application prototype. As

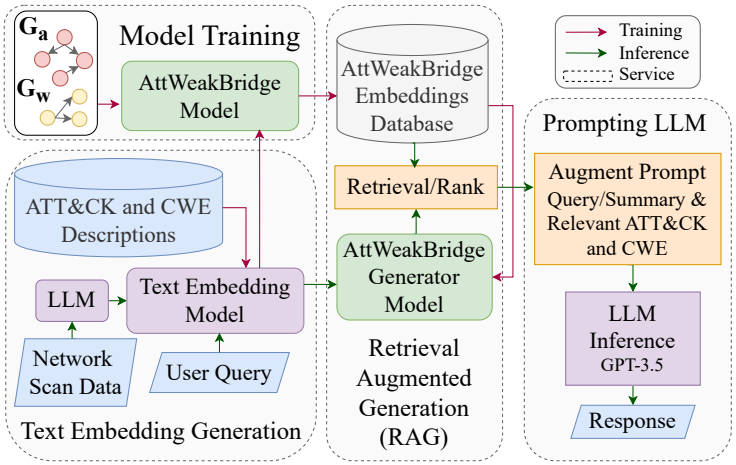


Fig. 1: AttWeakBridge application prototype pipeline.

shown in Figure 1, the system processes network scan data by summarizing it into text, extracting embeddings, and using retrieval-augmented generation (RAG) to identify relevant ATT&CK techniques and CWE weaknesses based on AttWeakBridge embeddings. The prototype then generates a security assessment. Further details are provided in Section 3.5. The key contributions of this paper are outlined below.

1. A Graph Contrastive Learning (GCL) approach tailored for graphs with two disconnected components, unlike traditional GCL methods.
2. A new positive-negative sampling technique for triplet creation in disconnected graphs, improving contrastive loss computation.
3. A novel inter-graph and intra-graph contrastive sampling strategy that integrates textual semantics with graph structure.
4. A dual-encoder GNN architecture that balances structural preservation and semantic alignment.
5. A complete application prototype showing how unified embeddings enable effective retrieval and knowledge discovery.

Data Background: This paper uses MITRE CWE and ATT&CK datasets, which represent abstractions of vulnerabilities and their respective attack patterns in certain domains. A vulnerability is a system weakness that can be exploited for unauthorized access or control. Since 1999, MITRE CVE has documented over 260,000 vulnerabilities. To manage this, CWE (939 nodes, 1,308 parent-child relationships) provides a higher-level abstraction, structured via the Research Concept View. MITRE ATT&CK models cyberattacks across domains like ICS (204 nodes, 781 edges) and enterprise networks (2,047 nodes, 19,436 edges). Both CWE and ATT&CK include textual descriptions. This work aims to align them in a shared mathematical space using textual and graph-based features.

2 Related Work

Learning representations over heterogeneous and disconnected graphs is a fundamental challenge in graph embedding and knowledge discovery. In many application domains, individual graph components capture complementary facets of a system but lack explicit links between them. This setting requires methods that can align nodes across graphs using weak or implicit signals, such as textual information. In this section, we review prominent state-of-the-art approaches in graph representation learning, with a particular focus on techniques designed for heterogeneous data.

2.1 Graph representation learning and contrastive methods

Graph Neural Networks (GNNs) have become the dominant paradigm for learning node embeddings by exploiting graph topology and node features [9]. However, GNNs inherently assume connectivity and struggle when graphs contain multiple disconnected components or when relationships across components are implicit rather than explicit.

Contrastive Learning (CL) has emerged as a powerful paradigm for unsupervised representation learning in vision and language [10, 11], and has been extended to graph domains as Graph Contrastive Learning (GCL) [12, 13]. Most GCL methods generate multiple augmented views of a single connected graph and maximize agreement between representations of the same node or subgraph across views. GraphCL [12] introduced stochastic graph augmentations such as node dropping and edge perturbation, while subsequent work refined augmentation strategies [14] or incorporated generative models to synthesize diverse views [15].

Despite their success, existing GCL methods are not designed for disconnected graph components. Augmentation-based contrastive objectives implicitly assume that positive pairs originate from the same graph instance, making them unsuitable for aligning nodes across separate graphs.

2.2 Bridging disconnected graph components

A small body of work has begun to address representation learning over multi-component graphs. Ma et al. [8] proposed introducing auxiliary edges between related nodes to connect components, but still treated nodes across components as negatives during contrastive learning. Such strategies risk biasing the embedding space or distorting the original graph structure.

Our proposed model, AttWeakBridge, directly targets the problem of aligning disconnected graph components without adding artificial edges or relying on labels. Our approach introduces a text-inspired GCL objective that leverages node text descriptions to guide inter-graph alignment, while preserving intra-graph structure through graph-based sampling. By combining inter-graph and intra-graph contrastive sampling with a dual-encoder GNN architecture, AttWeakBridge learns a unified embedding space for nodes across disconnected components. This positions our work as a principled advancement in graph contrastive learning for fragmented, text-rich graphs, with cybersecurity serving as a real-world instantiation of a broader data mining challenge.

2.3 Graph-based integration of cybersecurity knowledge

Within cybersecurity, efforts to relate attack patterns and weaknesses have been limited. The BRON framework [16] constructs a large cybersecurity knowledge graph by integrating multiple sources, including ATT&CK, CWE, CVE, and CAPEC. However, it relies heavily on manual labeling and supervised link prediction, limiting scalability and reproducibility. Other studies have explored supervised mappings between CVEs and ATT&CK techniques, particularly in industrial control system (ICS) settings [17, 18], but such approaches depend on costly annotations and are constrained to known relationships.

Unsupervised methods in this domain often rely on textual similarity between node descriptions [19, 20]. While effective for identifying semantically similar entities, text-only approaches ignore the rich structural information encoded in each graph, leading to incomplete or noisy alignments—an issue that generalizes beyond cybersecurity to many text-rich graph mining problems.

AttWeakBridge connects CWE and ATT&CK by aligning disconnected cybersecurity graphs without introducing artificial edges or relying on labeled links. By combining text-guided inter-graph contrastive learning with structure-preserving intra-graph sampling in a dual-encoder GNN framework, our method represents a fundamentally new direction for unsupervised cybersecurity knowledge integration.

3 Methodology

This section presents the AttWeakBridge framework, as illustrated in Figure 2. The subsequent subsections provide a detailed explanation of its components.

3.1 Problem Description

Let $D_a = (\tau^a, G_a)$ be the ATT&CK dataset containing $n_a = |\tau^a|$ ATT&CK patterns and a graph component G_a formed from the underlying relationships between $|\tau^a|$ ATT&CK nodes. Here, τ^a represents the set of text descriptions of all the ATT&CK nodes and G_a is a graph component with nodes $V_a = a_1, a_2, ..., a_{n_a}$ and edges E_a. The edges are represented as $(a_1, a_2) \in E_a$ where a_1 and a_2 has a relationships in D_a. Similarly, $D_w = (\tau^w, G_w)$ is the CWE weakness dataset containing text description of $n_w = |\tau^w|$ weaknesses, G_w being the graph that is composed of the 'child of' relationships between $|\tau^w|$ weakness nodes. G_w graph component includes nodes $V_w = w_1, w_2, ..., w_{n_w}$ and edges E_w. The edges are represented as $(w_1, w_2) \in E_w$ where w_1 and w_2 node has a 'child of' relationships in D_w. Both the graph components, G_a and G_w form the complete graph G, where the edge list is $E = (E_a \parallel E_w)$. Here, $\parallel$ refers to the concatenation of two arrays. T is a text embedding space, either fine-tuned or pre-trained, for all nodes' text description $(\tau^a \parallel \tau^w)$. Text descriptions of nodes from ATT&CK and CWE were encoded using GPT-2 embeddings. T_i refers to the text embedding of the i^{th} text description of $(\tau^a \parallel \tau^w)$. Here, τ^a is set of all the text from ATT&CK nodes, $(\tau^a{}_1, \tau^a{}_2, ...\tau^a{}_{n_a})$. Similarly, M_i refers to the node embedding of the i^{th} node in $(V_a \parallel V_w)$, where M is the node embedding space. Node embeddings for AttWeakBridge were generated using DeepWalk and node2vec.

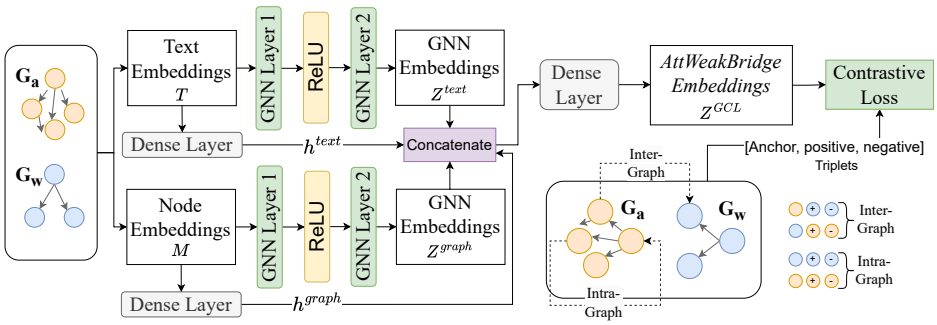


Fig. 2: AttWeakBridge architecture in creating a new embedding space through a contrastive loss to combine both graph features and textual information of nodes.

The objective is to learn a combined embedding space $Z = (z_{a_1}, ..., z_{a_{n_a}}, z_{w_1}, ..., z_{w_{n_w}})$ for all $V = (V_w \parallel V_a)$ nodes of the graph $G = (G_w, G_a)$, such that Z takes into account the text descriptions τ^w and τ^a, and graph structures E_w and E_a for the construction of the space.

3.2 Graph Neural Network

A Graph Neural Network (GNN) constructs an embedding space for all graph nodes. Given a graph $G = (V, E)$, graph representation learning aims to learn an encoder $f : V \rightarrow R^D$, such that the embeddings $f(v) \mid v \in V$ can be utilized for various downstream tasks, including node classification and link prediction. This paper introduces a specialized technique for graphs with two components and text-rich nodes. Our AttWeakBridge model uses GNNs as the encoder f where each node $v_i \in V$ $(V = (V_w \parallel V_a))$ starts with an initial representation $d^0_{v_i}$. The GNN iteratively updates node embeddings through neighborhood aggregation:

$$d^{(l)}_{v_i} = AGG(d^{(l-1)}_{v_i}, \{d^{(l-1)}_{v_j} | v_j \in N_{v_i}\}) \tag{1}$$

Here, $d^{(l)}_{v_i}$ represents the embedding of node v_i at l^{th} layer, N_{v_i} denotes the set of neighboring nodes of v_i, and AGG(.) is the aggregation function. For initialization, we define $d^0_{v_i}$ as either the text embedding T_i or the node embeddings M_i.

We refer to this conventional GNN as a single-encoder GNN because it takes one type of feature as its input. However, since the CWE and ATT&CK nodes have no direct edges, a conventional GNN trained with only graph-based features would likely keep them separate in the learned space, making it difficult to capture cross-graph relationships. To address this, we propose a dual encoder GNN explained in 3.3, which integrates both text embeddings T and node embeddings M. It employs two single encoder GNNs, one for each feature type, to generate the corresponding GNN embeddings Z^{text} and Z^{graph}. The GNN encoder layer train a graph convolutional network (GCN) or graph attention network (GAT) using a contrastive loss function, which requires positive and negative relationships (detailed in Subsection 3.4). A brief overview of GCN and GAT is provided in Appendix A.1.

3.3 Dual Encoder Graph Neural Network

The dual-encoder GNN architecture of AttWeakBridge integrates both the graph structure and textual descriptions of CWE and ATT&CK. It processes two feature sets—text and node embeddings—to construct a unified embedding space. As shown in Figure 2, the first GNN utilizes textual features of node descriptions denoted as text embeddings T and produce Z^{text}, while the second GNN leverages precomputed node embeddings M and produce Z^{graph}. To enhance representation, both embeddings, T and M pass through dense layers, producing $h^{\text{text}} = \text{Dense}(T)$ and $h^{\text{graph}} = \text{Dense}(M)$. The outputs, h^{text} and h^{graph}, from these dense layers, along with GNN-derived embeddings Z^{text} and Z^{graph}, are concatenated and processed through a final dense layer to generate the AttWeakBridge embeddings, denoted as Z^{GCL}. For the ith node, the embedding is:

$$Z_i^{\text{GCL}} = \sigma(W[h_i^{\text{text}} \parallel Z_i^{\text{text}} \parallel h_i^{\text{graph}} \parallel Z_i^{\text{graph}}] + b) \tag{2}$$

The entire model is optimized using a contrastive loss function detailed in the following subsection. This loss function maximizes the similarity between embeddings of similar nodes while minimizing the similarity between embeddings of dissimilar nodes, thereby enhancing the discriminative power of the learned embeddings.

3.4 Text-inspired Graph Contrastive Loss via Inter and Intra-Graph Selection of Samples

In this subsection, we introduce our proposed text-inspired graph contrastive loss function, which is designed to enhance graph representation learning by leveraging both textual information and graph structure of CWE and ATT&CK datasets. Traditional graph contrastive learning methods generate augmented views of graphs and select positive samples of the same node from different augmented views. However, this approach is ineffective for establishing relationships between disconnected graphs like CWE and ATT&CK because the triplets will miss all weakness-attack pattern pairs.

We could create new edges between CWE and ATT&CK nodes based on the text-similarity of the nodes. However, this approach might overlook the effect of each graph component thereby necessitating an approach that allows the optimization process to balance the trade-off between text and graph structural importance. Our proposed inter and intra-graph triplet selection overcomes these problems by utilizing both text and graph representations to construct triplets that better capture weakness-attack relationships, as detailed in the next subsection.

Inter-Graph and Intra-Graph Selection of Positive and Negative Samples for Triplet Construction: We aim to preserve the inherent graph structure information in the embeddings as well as bring the attack and weakness nodes to the same neighborhood. To achieve this, we incorporate both text-based similarities and graph-based similarities for positive and negative sampling. This approach ensures that the generated embeddings reflect both the textual content and the structural features of the nodes.

Inter-graph triplets: An anchor node is any node either from CWE or ATT&CK. If the anchor is from CWE, then both the negative and positive samples will be from

ATT&CK, and vice versa. Inter-graph triplets use only text embeddings, because there are no edges between CWE and ATT&CK. A positive sample against an anchor of a component is selected from highly textually similar nodes of the other component.

For node a_i (e.g., an attack node from graph $G_a, a_i \in V_a$), we compute the cosine similarity, $sim_{text}(a_i, w_j) = (a_i \cdot w_j)/\|a_i\|\|w_j\|$, with all nodes from different graph w_j (e.g., weakness nodes from graph G_w, $w_j \in V_w$).

Let S_i^{text} be the set of anchor, positive, and negative triplet for a node index i, such that:

$$S_i^{\text{text}} = \{(a_i, w_j^+, w_k^-) \mid \text{sim}_{\text{text}}(a_i, w_j) \geq \theta_{\text{pos-text}} \text{ and } \text{sim}_{\text{text}}(a_i, w_k) \leq \theta_{\text{neg-text}}\} \tag{3}$$

where, $\theta_{\text{pos-text}}$ and $\theta_{\text{neg-text}}$ are thresholds for selecting the top most similar and least similar nodes, respectively. $\text{sim}_{\text{text}}(x, y)$ computes the cosine similarity between two text embedding vectors.

Similarly, S_i^{text} can be formed for an anchor weakness w_i (where $w_i \in V_w$) using negative and positive samples from attacks a_j^+ and a_k^- (where $(a_j^+, a_k^-) \in V_a$). Note that the indices i, j, and, k refer to any position of nodes within the node set V.

To simplify threshold selection, our application automatically computes thresholds based on user-defined top and bottom percentiles. For example, with a 3% setting, the top 3% most similar CWE nodes to an attack pattern are positive samples, while the bottom 3% serve as negative samples.

Intra-graph triplets: To identify positive and negative nodes based on graph similarities, we compute the cosine similarity between the node embeddings of attack-attack nodes and weakness-weakness nodes. Note here that intra-graph triplets use node embeddings only, whereas the inter-graph triplets use text embeddings. This selection ensures the contrastive loss preserves the graph structure, as well as attempts to bring nodes of two graphs closer. Intra-graph triplets have the same threshold-based strategies for positive and negative samples, as inter-graph strategies, except that the intra-graph strategy creates positive and negative samples from the same component where the anchor is. Let S_i^{graph} be the set of anchor, positive, and negative triplet for a node index i, such that:

$$S_i^{\text{graph}} = \{(a_i, a_j^+, a_k^-) \mid \text{sim}_{\text{graph}}(a_i, a_j) \geq \theta_{\text{pos-graph}} \text{ and } \text{sim}_{\text{graph}}(a_i, a_k) \leq \theta_{\text{neg-graph}}\} \tag{4}$$

Similarly, S_i^{graph} can be formed for an anchor weakness w_i using negative and positive samples from weakness w_j^+ and w_k^-.

AttWeakBridge's Contrastive Loss Functions: The contrastive loss function in AttWeakBridge enforces a balanced embedding space, ensuring that nodes similar in graph structure or text semantics obtain similar representations. When attack nodes from V_a are used as anchor nodes, the loss is defined as:

$$\mathcal{L}_{\text{attack-anchor}} = \sum_{i \in V_a} \left(\sum_{(w_j^+, w_k^-) \in S_i^{text}} \left| sim(z_{a_i}, z_{w_k^-}) - sim(z_{a_i}, z_{w_j^+}) + \delta_t \right| + \sum_{(a_j^+, a_k^-) \in S_i^{graph}} \left[sim(z_{a_i}, z_{a_k^-}) - sim(z_{a_i}, z_{a_j^+}) + \delta_g \right] \right) \tag{5}$$

Similarly, when weakness nodes serve as anchor nodes, the corresponding loss function is defined as:

$$\mathcal{L}_{\text{weakness-anchor}} = \sum_{i \in V_w} \left(\sum_{(a_j^+, a_k^-) \in S_j^{text}} \left[sim(z_{w_i}, z_{a_k^-}) - sim(z_{w_i}, z_{a_j^+}) + \delta_t \right] + \sum_{(w_j^+, w_k^-) \in S_j^{graph}} \left[sim(z_{w_i}, z_{w_k^-}) - sim(z_{w_i}, z_{w_j^+}) + \delta_g \right] \right) \tag{6}$$

$$\mathcal{L}_{total} = \mathcal{L}_{\text{attack-anchor}} + \mathcal{L}_{\text{weakness-anchor}} \tag{7}$$

Here, $\mathcal{L}_{\text{total}}$ represents the total loss. $\text{sim}(z_x, z_y)$ measures the cosine similarity between two embedding vectors from the embedding space, Z^{GCL}, created by AttWeakBridge. δ_t and δ_g are hyperparameters for margin ensuring a minimum difference between positive and negative pairs.

3.5 Application Prototype: Generate AttWeakBridge Embeddings from Text

This section outlines the process of generating AttWeakBridge embeddings from textual data and integrating them into a retrieval-augmented application. Figure 1 illustrates the pipeline.

The application supports two types of inputs: natural language prompts and network scan data. For scan data, a Large Language Model (LLM) summarizes it into a structured prompt. To retrieve prompt relevant ATT&CK and CWE nodes, input prompts must be converted into AttWeakBridge embeddings. Since generating these embeddings directly requires textual and graph data—unavailable for arbitrary inputs—we trained an embedding generator model to transform any text descriptions into AttWeakBridge embeddings. This model was trained using ATT&CK and CWE node's text embeddings and corresponding AttWeakBridge embeddings. We explored two architectures: an Encoder-Decoder model and a Generative Adversarial Network (GAN), both with three hidden layers (512, 256, 128) and trained for 1000 epochs.

For retrieval-augmented generation, the prompt is first converted into a text embedding using GPT-2, then passed through the generator to obtain its AttWeakBridge embedding. The generated prompt embeddings are compared with the precomputed AttWeakBridge embeddings of ATT&CK and CWE nodes and relevant nodes are retrieved based on similarity metrics. These retrieved descriptions are appended to the prompt, enhancing its context before being sent to an LLM for generating a context-aware and informative response. Figure 1 provides a comprehensive view of the pipeline.

4 Experimental Results

In this section, we extensively evaluate our AttWeakBridge models and a downstream application. The questions we seek to answer are as follows:

1. How effectively does AttWeakBridge integrate and leverage both graph and text properties? (Subsection 4.1)
2. How well does AttWeakBridge bring attack and weakness nodes together in the embedding space? (Subsection 4.2)
3. How does the quality of the AttWeakBridge's embedding space react to different settings? (Subsection 4.3)
4. How does AttWeakBridge perform compared to traditional graph contrastive models that rely on conventional view augmentation techniques? (Subsection 4.4)

Datasets: We evaluated the performance of AttWeakBridge on three real-world cybersecurity datasets. The ATT&CK dataset includes MITRE ICS ATT&CK [21], MITRE Enterprise ATT&CK [22]. The weaknesses graph is built using MITRE CWE dataset [23]. Additionally, two augmented datasets were included to assess scalability. We summarize the statistics of all the datasets in Table 1.

Table 1: Datasets.

Dataset Name	# Nodes	# Edges
MITRE ICS ATT&CK	204	781
MITRE Enterprise ATT&CK	2,047	19,436
MITRE CWE	939	1,308
Augmented ATT&CK	258,525	2,241,796
Augmented CWE	56,478	132,108

Parameter settings and runtime[1]**:** The GNNs (GCN or GAT) have two hidden layers with dimensions 256 and 128. Models are trained using the Adam optimizer (learning rate 0.001). GPT-2 is fine-tuned for 10 epochs, while GNNs train for 400 epochs. Experiments run on an RTX A6000 GPU, with AttWeakBridge (dual-encoder, CWE & Enterprise ATT&CK) completing training in under 25 seconds for 400 epochs.

[1] AttWeakBridge source code will be released on GitHub after publication.

Real-world ATT&CK and CWE datasets are small, so we used a GAN to generate augmented graphs, 100× larger than the original. On this augmented graph, text embedding (GPT-2) generation took 10 minutes, node embeddings (DeepWalk) 20 minutes, and triplet construction (1.5M triplets) under 12 hours. Model training finished in 12 minutes (20 epochs). Subsection 4.5 includes a table detailing AttWeakBridge training times for different type of input features and GNN models.

4.1 How Effectively does AttWeakBridge Utilize both Graph and Text Properties?

AttWeakBridge focuses on bringing nodes with similar text and graph properties within the same neighborhood. In this experiment, we investigate how effectively AttWeakBridge brings such nodes together. Since the experiment is focused on the neighborhood, we use precision@k, where k is the number of nearest neighbors. The precision is computed with respect to either graph neighborhood using shortest path distance, or text distance using BERT embeddings. Figure 3 compares eight variations of the AttWeakBridge model, including text vs. graph emb., GCN vs. GAT, and single vs. dual encoder. The dual encoder settings use both DeepWalk and finetuned GPT-2 embeddings.

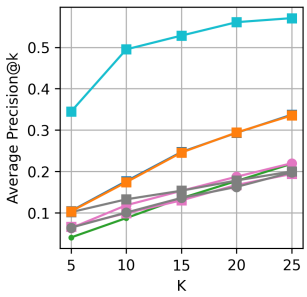
Figure 3a and 3b are for ICS ATT&CK nodes with precision denominator of graph and text, respectively. Figure 3c and 3d are for CWE nodes with precision denominator of graph and text respectively. Figure 3e provides the legends.

Notice that for both Figure 3a and 3c, the DeepWalk embeddings (cyan line) has the highest average precision for any k of kNN, which is to be expected because the precision is computed against the shortest path distance-based nearest neighbors. Theoretically, the line for fine-tuned GPT-2 text embeddings (green line) should have the lowest precision in 3a and 3c given that these two plots are computing precision against graph properties. The green line (fine-tuned GPT-2) is evident to have low precision. Their single and dual encoder combinations perform reasonably in most cases better than GPT-2 embeddings for ATT&CK and in dual-encoder cases in weaknesses.

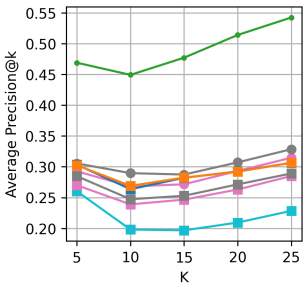
Figure 3b and 3d contain average precision with respect to kNN using BERT embeddings. Expectedly, the cyan line for graph-based DeepWalk tends to exhibit lower precision against the text-denominator precision. Most single and dual-encoder settings (pink, gray, blue, orange lines) in 3a and 3c have higher precision than fine-tuned GPT-2 text embedding (green). Most single and dual-encoder settings in 3b and 3d have higher precision than DeepWalk (cyan).

With Figure 3a, 3b, 3c, and 3d, the common trend is that the single and dual encoder-based AttWeakBridge has a balanced precision higher than BERT-based precision when DeepWalk is the precision-denominator, and higher than DeepWalk-based precision when BERT is the precision-denominator. The observations indicate that AttWeakBridge creates a balance between text and node embeddings.

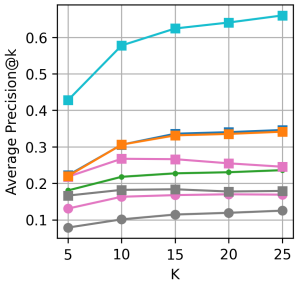
Now, **which one is better for AttWeakBridge, single encoder or dual encoder?** While all the single and dual encoder settings (pink and gray lines) incorporate text and graph properties in different ways, the plots indicate that the dual encoder settings (blue and orange lines) outperform the single encoder settings in terms of precision. In the plots, the dual encoder GAT (orange lines) overlaps with the



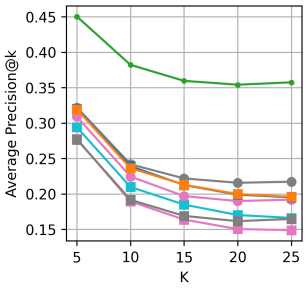
(a) ATT&CK node's kNN ∩ kNN using shortest path distance



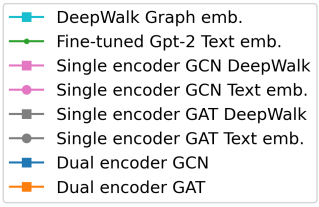
(b) ATT&CK node's kNN ∩ BERT-based kNN



(c) CWE node's kNN ∩ kNN using shortest path distance



(d) CWE node's kNN ∩ BERT-based kNN



(e) Legend shows different variations of the AttWeakBridge model

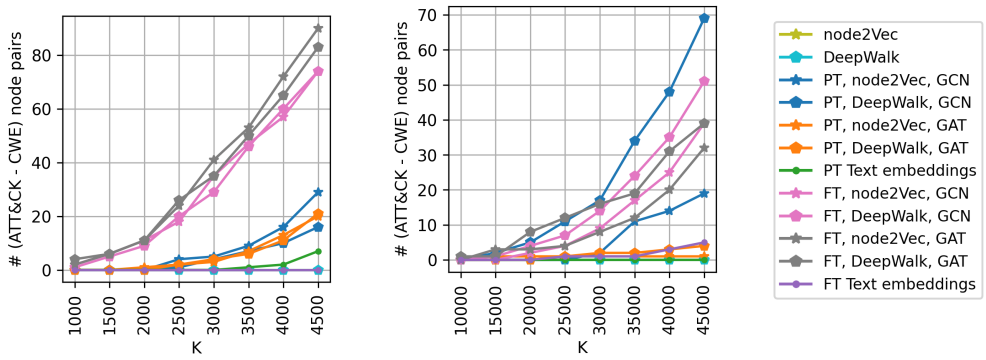
Fig. 3: Precision@k results for ICS ATT&CK (a, b) and CWE (c, d). AttWeakBridge models use single or dual encoders with GCN or GAT. Single encoders rely on either graph (DeepWalk) or text embeddings, while dual encoders combine both. Results are compared against DeepWalk and GPT-2 embeddings.

dual encoder GCN most of the time, hiding the blue line for the dual encoder GCN. The orange line (dual encoder GAT) is consistently better than any single encoder setting for the 3a and 3c. For 3b and 3d, the single-encoder GAT (text) has slightly better precision than the dual-encoder GAT. The precision results of the single encoder GAT (text) in 3a and 3c are one of the worst. Dual encoder GAT (orange) and dual encoder GCN (blue line occluded by the orange line) have consistent balance in all of the plots 3a, 3b, 3c, and 3d, making the dual encoder setting the preferred general setting for AttWeakBridge.

The reason AttWeakBridge with dual encoders performs better than single encoders is that it simultaneously learns graph and text properties in two encoders in addition to its contrastive loss mechanism, whereas a single encoder setting has lesser scope and learnable parameters in the architecture to capture intricate graph and text properties.

4.2 How Well does AttWeakBridge Bring Nodes from Tow Graph Components Closer in the Embedding Space?

AttWeakBridge aims to bring CWE and ATT&CK nodes closer when they have high text similarity while preserving graph structures. The more such attack-weakness pairs are discovered, the more useful the system will be from a cybersecurity perspective.



(a) ICS ATT&CK-CWE. **(b)** Ent. ATT&CK-CWE **(c)** Legends.

Fig. 4: Number of ATT&CK-CWE pairs among the top-K most similar node pairs in AttWeakBridge's embedding space. The comparison includes vanilla graph (node2vec, DeepWalk) and text embeddings (pretrained and fine-tuned GPT-2). AttWeakBridge, combining text and graph embeddings with GCN or GAT, effectively identifies cross-graph pairs, unlike baseline models.

Figure 4 shows the number of attack-weakness pairs (CWE and ICS ATT&CK) among the top-k most similar node pairs in AttWeakBridge's embedding space. PT refers to pretrained GPT-2, and FT to fine-tuned GPT-2. Except for node2vec, DeepWalk, PT, and FT text embeddings, all models use AttWeakBridge's dual encoder. Figure 4 shows that baseline models relying only on graph (node2vec, DeepWalk) or text embeddings (PT, FT) failed to detect inter-graph connections within the top 4500 pairs. AttWeakBridge with pretrained GPT-2 embeddings captured some ATT&CK-CWE pairs within the top 2500, while fine-tuned GPT-2 embeddings significantly improved pair detection.

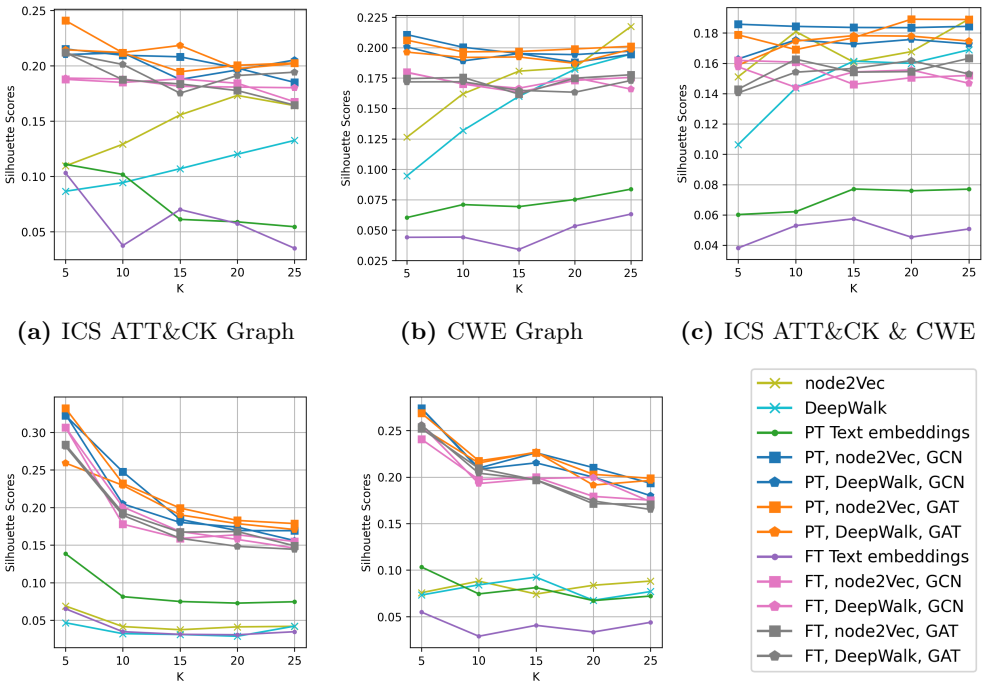
These results show that while fine-tuned GPT-2 embeddings alone do not align ATT&CK-CWE pairs, AttWeakBridge's dual encoder effectively integrates text and graph features to bring them closer, bridging the two graph components.

Figure 4b shows results for Enterprise ATT&CK, aligning with ICS ATT&CK findings. Due to the larger size of this dataset, the plot spans a wider K range. Notably, AttWeakBridge with pretrained text embeddings identifies more ATT&CK-CWE pairs than fine-tuned embeddings, likely because the fine-tuning potentially limits the generalization capability of text embeddings.

4.3 Impact of Settings on AttWeakBridge's Embedding Quality

This subsection evaluates the embedding quality of AttWeakBridge against text-only (PT, FT GPT-2) and graph-only (node2vec, DeepWalk) models. Figure 5 shows the Average Silhouette Score (ASC) for K-means clustering on ATT&CK (Figure 5a), CWE (Figure 5b), and combined embeddings (Figure 5c), where higher scores indicate better clusters.

PT text embeddings (green line) and FT text embeddings (a purple line at the bottom of each plot) consistently have low ASCs. node2vec and DeepWalk tend to have lower ASCs with a lower number of clusters and an increasing trend with more clusters (larger K). Having many clusters is not useful in most cases because the idea of clustering is to obtain the analytical benefit of having similar elements in a small number of groups. All dual encoder settings of AttWeakBridge consistently perform better than the vanilla embeddings in terms of ASC. ASCs of AttWeakBridge with pre-trained GPT-2 as input embeddings are slightly better than those of AttWeakBridge



(a) ICS ATT&CK Graph **(b)** CWE Graph **(c)** ICS ATT&CK & CWE

(d) Ent. ATT&CK Graph **(e)** Ent. ATT&CK & CWE **(f)** Legends

Fig. 5: Average Silhouette Scores at different number of clusters K. The AttWeakBridge variations are trained on the pre-trained (PT.) or fine-tuned (FT.) text embeddings with DeepWalk or node2vec graph embeddings using GCN or GAT for learning. AttWeakBridge variations are compared with vanilla graph embeddings (DeepWalk and node2vec) and pre-trained (PT.) and fine-tuned (FT.) text embeddings.

with fine-tuned GPT-2 embeddings. However, given the experiment in Section 4.2 showed that AttWeakBridge with fine-tuned GPT-2 input captured more ATT&CK-CWE relationships, the choice between a fine-tuned or pre-trained text embedding model for AttWeakBridge should depend on the specific application objective.

Figure 5d and 5e presents the results for Enterprise ATT&CK data. These plots also demonstrate the superior performance of AttWeakBridge's dual encoder settings over vanilla embeddings in terms of Average Silhouette Score.

4.4 Supervised Evaluation of AttWeakBridge

There is no labeled relationship between CWE and ATT&CK nodes. If there were one, we could use that to train a simple classification model to create edges between their nodes. There are existing research efforts that connected CWE and another dataset named CAPEC (Common Attack Pattern Enumerations and Classifications). Some other efforts connected CAPEC and ATT&CK. To create a supervised mechanism for the evaluation of AttWeakBridge, we established relationships between CWE and ATT&CK nodes via CWE-CAPEC and CAPEC-ATT&CK relationships as suggested by BRON [16].

The evaluation of BRON-dependent relationships is not comprehensive, which limits their usability for training machine learning models. Nonetheless, we carried out this experiment to determine the extent to which our AttWeakBridge automatically identifies BRON-dependent relationships, despite not being aware of CAPEC. We developed view-based augmentations for Graph Contrastive Learning (GCL) to evaluate and compare them with our AttWeakBridge model. To create views for each attack node, we calculated the top five percent weakness nodes based on text embedding similarities and added edges between the attack and most similar weakness nodes. We refer to this augmented graph as Text_{aug}. This experiment uses four variations of the original graph and the augmented graph, Text_{aug}. We name these four variations View Aug. 1 through 4. The first variation creates two views from the original graph

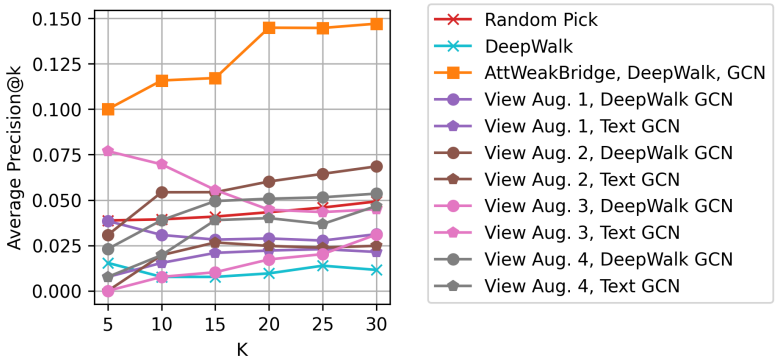


Fig. 6: Precision@k of AttWeakBridge vs View Augmentation compared to BRON-Graph. Dataset: ICS ATT&CK-CWE.

utilizing edge-dropping. The second variation creates two views of Text_{aug} by edge-dropping. The third variation creates one view from the original graph and another view from Text_{aug}. The fourth variation directly uses the original graph and Text_{aug} as two views.

Figure 6 shows average Precision@k where the denominator is BRON-based nearest neighbors at different k. AttWeakBridge (the orange line at top) exhibits the maximum average precision at any k in Figure 6, indicating that AttWeakBridge outperforms any view-driven approaches of GCL using either graph or text embeddings. Also, AttWeakBridge outperforms vanilla DeepWalk and random pick of nearest neighbors. Notice that some of the view-based GCL approaches perform even worse than the random selection of nearest neighbors.

4.5 Runtime Analysis

Table 2 provides runtime for training of the dual-encoder AttWeakBridge, which is the most complex setting using different features. With the Enterprise ATT&CK data and CWE, the training time is less than 25 seconds for 400 epochs. The augmented data is trained over 10 epochs, utilizing only pre-trained GPT-2 embeddings. Fine-tuning is not applied, as the augmented data comprises synthesized descriptions rather than real-world text. A single RTX A6000 GPU is used in all experiments.

The time taken for the creation of the models is an amortized initial cost. Given a prompt, the entire inference (prompt to retrieval using RAG) takes less than a second.

Table 2: Training runtime for different variations of Dual Encoder AttWeakBridge on ICS, Enterprise, and Augmented Dataset.

			Runtime (sec)		
LLM	**GNN Model**	**Node Feature**	**ICS**	**Enterprise**	**Augmented Graph**
Pretrained GPT-2	GCN	node2vec	5.77	21.17	328.22
		DeepWalk	5.95	20.85	325.12
	GAT	node2vec	6.81	24.02	369.12
		DeepWalk	7.14	24.09	368.35
Finetuned GPT-2	GCN	node2vec	24.44	21.18	NA
		DeepWalk	24.03	21.17	NA
	GAT	node2vec	23.50	23.94	NA
		DeepWalk	24.24	24.09	NA

4.6 Case Study: LLM Assisted Security Analysis

This subsection presents a case study evaluating the application prototype from Subsection 3.5. The 2021 Colonial Pipeline attack, caused by the DarkSide ransomware

group, resulted in major fuel shortages across the Eastern U.S. due to a compromised VPN password without multi-factor authentication.

Using a query inspired by this cyberattack, AttWeakBridge retrieved relevant ATT&CK techniques (e.g., T0822: External Remote Services) and CWE weaknesses (e.g., CWE-256: Plaintext Password Storage), closely aligning with the Colonial Pipeline cyberattack. The application uses these retrieved ATT&CK and CWE descriptions to augment the input, improving the LLM-generated security recommendations.

With augmentation, the LLM suggested key mitigations such as Multi-Factor Authentication (MFA) and enforcing strong password policies, whereas without augmentation, it only advised against saving passwords locally. This case study highlights AttWeakBridge's effectiveness in enhancing cybersecurity responses through retrieval-augmented generation (RAG). Additional sample queries and results are presented in the subsequent case studies.

Case Study 1 (Colonial Pipeline Attack): In 2021, the Colonial Pipeline, a major fuel pipeline system in the United States, was hit by a ransomware attack by the DarkSide ransomware group [24]. The attackers breached the pipeline's IT systems, encrypting critical data and prompting a precautionary shutdown. The attack triggered widespread fuel shortages and disruptions across the Eastern U.S.

Investigations revealed that the Colonial Pipeline attackers used a compromised password of a VPN account, which did not have multi-factor authentication [24]. For this case study, inspired by the Colonial Pipeline Attack scenario, we form a statement that a network administrator may provide, which we use as a query for AttWeakBridge.

As detailed in Section 3.5 the AttWeakBridge application prototype supports two input types: natural language prompts and network scan data. In this case study, we utilized a natural language prompt inspired by the Colonial Pipeline Attack. The input statement (prompt) provided to the prototype is: "Users access computers remotely using VPN. They save their passwords on the computers they use to log in remotely." Ideally, a network administrator will enter such a statement to the AttWeakBridge prototype.

The retrieval augmented generation retrieve ten ATT&CK among which top three are:

- T0822 [2]: Adversaries may leverage external remote services as a point of initial access into your network.
- M0801 [3]: Access Management technologies can be used to enforce authorization polices and decisions...
- M0932 [4]: Use two or more pieces of evidence to authenticate to a system; such as username and ...

It also returned ten CWE nodes descriptions; among them, the top three are:

- CWE 261 [5]: Weak Encoding for Password. Passwords should be encrypted with keys that are at least 128..
- CWE 603 [6]: Use of Client-Side Authentication. Do not rely on client side data. Always perform server side...

- CWE 256 [7]: Plaintext Storage of a Password. A programmer might attempt to remedy the password...

The description of the above entries demonstrates that the top relevant ATT&CK and CWE nodes identified by our RAG framework for the given input statement are closely related to the context of the Colonial Pipeline attack. Notably, mitigation M0801, which is the second top ATT&CK node suggested by our RAG, is recognized as one of the most effective strategies to mitigate attacks similar to the Colonial Pipeline incident [24].

By augmenting the user-provided input with the top CWE and ATT&CK node descriptions retrieved by our AttWeakBridge prototype, the system produced an enhanced prompt for LLM (GPT-3.5Turbo). In response to the augmented prompt, LLM generated the following security recommendations:

Based on the information provided in the prompt and related attack and weakness descriptions, here are some recommendations to improve the security of remote access using VPN and password management:

1. *Multi-Factor Authentication (MFA): Implement MFA for remote access to add an extra layer of security beyond passwords. .... (cropped for brevity)*
2. *Enforce Secure Password Policies: Ensure that strong password policies are enforced, such as minimum length, complexity requirements, and regular password changes. This helps in preventing weak and easily guessable passwords .... (cropped for brevity)*

Here, we can observe that the top suggestion by LLM when given the augmented prompt is to use Multi-Factor Authentication (MFA), and the second top suggestion is to enforce secure password policies. In contrast, when LLM (GPT-3.5Turbo) was provided only the user-provided statement without the augmented information, it returned a simpler recommendation:

It is not recommended to save passwords on computers that are being used to connect remotely via VPN. Saving passwords in this manner can pose a security risk, especially if the computer is lost, stolen, or accessed by unauthorized individuals. Instead, it is recommended to use a secure password manager to store and manage passwords securely. This way, users can access their passwords when needed without compromising security.

This comparison highlights the effectiveness of our prototype in identifying contextually relevant issues and generating enriched prompts. By leveraging CWEs and ATT&CK techniques retrieved from the AttWeakBridge embeddings, the system enhances the prompt to provide actionable, comprehensive mitigation strategies tailored to the given scenario.

[2] MITRE Corporation ATT&CK T0822, Link: https://attack.mitre.org/techniques/T0822
[3] MITRE Corporation ATT&CK M0801, Link: https://attack.mitre.org/techniques/M0801
[4] MITRE Corporation ATT&CK M0932, Link: https://attack.mitre.org/techniques/M0932
[5] MITRE Corporation CWE-261, Link: https://cwe.mitre.org/data/definitions/261.html
[6] MITRE Corporation CWE-603, Link: https://cwe.mitre.org/data/definitions/603.html
[7] MITRE Corporation CWE-256, Link: https://cwe.mitre.org/data/definitions/256.html

Case Study 2 (NetScan Data): In this case study, we demonstrate the application of AttWeakBridge in analyzing network vulnerability scan data related to CVE-2021-41773 (Path Traversal Vulnerability). A sample network vulnerability scan is shown in Figure 7.

80/tcp open http Apache httpd 2.4.6
— http-vuln-cve-2021-41773:
— VULNERABLE:
— Apache HTTP Server Path Traversal Vulnerability
(CVE-2021-41773)
— State: VULNERABLE
— Risk Factor: CRITICAL
— This vulnerability allows a remote attacker to traverse directories via a crafted request.
— Disclosure date: 2021-09-16

Fig. 7: Sample network scan data detecting CVE-2021-41773.

Using this scan data as input, GPT-3.5Turbo generated the following summary:

The scan detected an open HTTP service on port 80/tcp, running Apache httpd version 2.4.6, which is vulnerable to the Path Traversal Vulnerability (CVE-2021-41773). This critical vulnerability allows remote attackers to traverse directories using specially crafted HTTP requests, potentially exposing sensitive files and system information. The issue was disclosed on September 16, 2021, and poses a high-security risk. Immediate action is recommended to mitigate this vulnerability by updating Apache to a patched version.

This summary was processed through the AttWeakBridge embedding generator (detailed in Section 3.5) to produce AttWeakBridge embeddings equivalent to the text. These embeddings were used in the application's Retrieval-Augmented Generation (RAG) process to retrieve relevant ATT&CK and CWE nodes. The RAG retrieved the following nodes:

- ATT&CK nodes: T1204.002 [8], T1036.005 [9], T1027.013 [10]
- CWE nodes: CWE-36 [11], CWE-943 [12], CWE-228 [13]

Among the retrieved nodes, CWE-36 (Absolute Path Traversal) is directly correlates with CVE-2021-40438, as it accurately describes the vulnerability's mechanism. From the top ATT&CK nodes, T1036.005 (Match Legitimate Name or Location) is particularly relevant. After exploiting the path traversal vulnerability to gain unauthorized server access, an attacker may place malicious files in directories typically

[8]MITRE Corporation ATT&CK T1204.002, Link:https://attack.mitre.org/techniques/T1204/002
[9]MITRE Corporation ATT&CK T1036.005, Link:https://attack.mitre.org/techniques/T1036/005
[10]MITRE Corporation ATT&CK T1027.013, Link:https://attack.mitre.org/techniques/T1027/013
[11]MITRE Corporation CWE-36, Link:https://cwe.mitre.org/data/definitions/36.html
[12]MITRE Corporation CWE-943, Link:https://cwe.mitre.org/data/definitions/943.html
[13]MITRE Corporation CWE-228, Link:https://cwe.mitre.org/data/definitions/228.html

associated with legitimate use or assign filenames that mimic legitimate files to evade detection.

This case study demonstrates the ability of AttWeakBridge to effectively map real-world vulnerabilities, such as CVE-2021-41773, to their underlying CWE weaknesses and ATT&CK patterns. By processing network scan data, the framework provides valuable insights into potential system weaknesses and attack strategies. Such functionality highlights the utility of AttWeakBridge in leveraging netscan data for enhanced threat analysis, enabling better-informed mitigation strategies and proactive defense measures.

5 Conclusions

This paper introduced AttWeakBridge, a novel Graph Contrastive Learning (GCL) framework for learning unified representations over disconnected, text-rich graph components. By employing a dual-encoder GNN architecture, AttWeakBridge jointly models structural dependencies within graph components and semantic information from node text, enabling effective alignment in a shared embedding space. We further proposed an intra- and inter-graph contrastive sampling strategy that enables contrastive learning across disconnected graphs without introducing artificial edges or relying on labeled links.

Extensive experimental analysis demonstrates that AttWeakBridge achieves a favorable balance between textual and structural signals, outperforming text-only and graph-only baselines in similarity learning and cross-component retrieval. The results also highlight the importance of combining contrastive objectives with multi-view encoders when learning representations over fragmented graphs. While our evaluation is grounded in real-world cybersecurity knowledge graphs (MITRE ATT&CK and CWE), the proposed framework is domain-agnostic and broadly applicable to data mining and knowledge discovery tasks involving heterogeneous, disconnected graph data with textual attributes.

Finally, we demonstrated the practical utility of the learned embeddings through a retrieval-augmented prototype, illustrating how unified graph representations can support downstream reasoning and decision-making tasks. Future work will explore extending this framework to additional graph modalities and large-scale heterogeneous graph settings.

6 Declarations

Competing Interest: The authors declare no competing interests.
Funding: This material is based upon work partially supported by the U.S. Department of Energy's Office of Energy Efficiency and Renewable Energy (EERE) under the Advanced Materials and Manufacturing Technologies Office, Award Number DEEE0009046. The views expressed herein do not necessarily represent the views of the U.S. Department of Energy or the United States Government.

7 Data Availability

The datasets supporting the findings of this study are publicly available from the following sources:

MITRE ATT&CK Framework Data: The MITRE ATT&CK datasets (Enterprise, ICS, and Mobile) are publicly available at https://attack.mitre.org/resources/attack-data-and-tools/.

Common Weakness Enumeration (CWE) Data: The CWE dataset (version 4.12) is publicly available at https://cwe.mitre.org/data/index.html.

Code and Reproducibility: All data preprocessing scripts, model implementations, and analysis code are publicly available at https://github.com/Rownak/AttWeakBridge-GCL. Processed datasets, embeddings, and model outputs can be reproduced by following the execution pipeline documented in the repository. Complete instructions for reproducing all experiments are provided in the repository's README file.

A APPENDICES

A.1 Graph Convolution Network and Graph Attention Network

In the following, we provide a brief overview of the GCN and GAT models.

Graph Convolutional Neural Network (GCN): GCN [9] is designed to capture the graph structure and node features for the embedding space Z by aggregating features from neighboring nodes. The GCN layer is defined as:

$$H^{(l+1)} = \sigma(\tilde{D}^{-1/2}\tilde{A}\tilde{D}^{-1/2}H^{(l)}W^{(l)}) \tag{8}$$

where, $H^{(l)}$ is the node feature matrix at layer l, initialized as $H^{(0)} = X_v$ (text embeddings T or node embeddings M). $\tilde{A} = A + I$ is the adjacency matrix with added self-loops with A as the adjacency matrix and I as the identity matrix. $\tilde{D}$ is the degree matrix of $\tilde{A}$. $W^{(l)}$ is the learnable weight matrix and σ is an activation function, (e.g., ReLU). The final GCN embeddings are $Z = H^{(L)}$, where L is the number of GCN layers.

Graph Attention Neural Network (GAT): The Graph Attention Neural Network (GAT) [25] introduces attention mechanisms to weigh the importance of neighboring nodes' features, allowing the model to focus on the most relevant parts of the graph. The GAT layer is defined as:

$$h_i^{(l+1)} = \sigma\left(\sum_{j \in N_i} \alpha_{ij}^{(l)} W^{(l)} h_j^{(l)}\right) \tag{9}$$

where, $h_i^{(l)}$ is the feature vector of node i at layer l. $W^{(l)}$ is the learnable weight matrix for layer l. $\alpha_{ij}^{(l)}$ is the attention coefficient for the edge (i, j), indicating the importance of node j's features to node i. The attention coefficients $\alpha_{ij}^{(l)}$ are computed as:

$$\alpha_{ij}^{(l)} = \frac{\exp\left(\text{LeakyReLU}\left(\vec{a}^{(l)T}[W^{(l)}h_i^{(l)} \parallel W^{(l)}h_j^{(l)}]\right)\right)}{\sum_{k \in N_i} \exp\left(\text{LeakyReLU}\left(\vec{a}^{(l)T}[W^{(l)}h_i^{(l)} \parallel W^{(l)}h_k^{(l)}]\right)\right)} \tag{10}$$

where, $\vec{a}^{(l)}$ is the learnable attention vector at layer l and $\parallel$ denotes concatenation. LeakyReLU is the Leaky Rectified Linear Unit activation function. The final GAT embeddings are $Z = \left(h_1^{(L)}, h_2^{(L)}, ..., h_{|V|}^{(L)}\right)$, where L is the number of GAT layers. In AttWeakBridge, both GCN and GAT use either text or node embeddings as initial features.

References

[1] Petsinis, P., Skitsas, K., Ranu, S., Mottin, D., Karras, P.: Alpine: Partial unlabeled graph alignment. In: Proceedings of the 31st ACM SIGKDD Conference on

Knowledge Discovery and Data Mining V.2. KDD '25, pp. 2315–2325. Association for Computing Machinery, New York, NY, USA (2025). https://doi.org/10.1145/3711896.3736839 . https://doi.org/10.1145/3711896.3736839

[2] Yan, H., Li, C., Long, R., Yan, C., Zhao, J., Zhuang, W., Yin, J., Zhang, P., Han, W., Sun, H., Deng, W., Zhang, Q., Sun, L., Xie, X., Wang, S.: A comprehensive study on text-attributed graphs: benchmarking and rethinking. In: Proceedings of the 37th International Conference on Neural Information Processing Systems. NIPS '23. Curran Associates Inc., Red Hook, NY, USA (2023)

[3] Zhou, C., Wang, Z., Chen, S., Du, J., Zheng, Q., Xu, Z., Huang, X.: Taming language models for text-attributed graph learning with decoupled aggregation. In: Che, W., Nabende, J., Shutova, E., Pilehvar, M.T. (eds.) Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 3463–3474. Association for Computational Linguistics, Vienna, Austria (2025). https://doi.org/10.18653/v1/2025.acl-long.173 . https://aclanthology.org/2025.acl-long.173/

[4] Yu, Z., Jin, D., Wei, J., Li, Y., Liu, Z., Shang, Y., Han, J., Wu, L.: Teko: Text-rich graph neural networks with external knowledge. IEEE Transactions on Neural Networks and Learning Systems **35**(10), 14699–14711 (2024) https://doi.org/10.1109/TNNLS.2023.3281354

[5] Fazaeli, M., Momtazi, S.: Node classifications with djcane: Disjoint content and network embedding. J. Inf. Sci. **50**(4), 977–990 (2024) https://doi.org/10.1177/01655515221111002

[6] 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 **32**(1), 4–24 (2021) https://doi.org/10.1109/TNNLS.2020.2978386

[7] Corso, G., Stark, H., Jegelka, S., Jaakkola, T., Barzilay, R.: Graph neural networks. Nature Reviews Methods Primers **4**(1) (2024) https://doi.org/10.1038/s43586-024-00294-7

[8] Ma, Y., Song, Z., Hu, X., Li, J., Zhang, Y., King, I.: Graph component contrastive learning for concept relatedness estimation. AAAI **37**(11), 13362–13370 (2023)

[9] Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. In: International Conference on Learning Representations (ICLR) (2017)

[10] Le-Khac, P.H., Healy, G., Smeaton, A.F.: Contrastive representation learning: A framework and review. IEEE Access **8**, 193907–193934 (2020)

[11] Jaiswal, A., Babu, A.R., Zadeh, M.Z., Banerjee, D., Makedon, F.: A survey on contrastive self-supervised learning. Technologies **9**(1) (2021)

[12] You, Y., Chen, T., Sui, Y., Chen, T., Wang, Z., Shen, Y.: Graph contrastive learning with augmentations. Advances in neural information processing systems **33**, 5812–5823 (2020)

[13] You, Y., Chen, T., Shen, Y., Wang, Z.: Graph contrastive learning automated. In: International Conference on Machine Learning, pp. 12121–12132 (2021). PMLR

[14] Zhu, Y., Xu, Y., Yu, F., Liu, Q., Wu, S., Wang, L.: Graph contrastive learning with adaptive augmentation. In: Proceedings of the Web Conference 2021. WWW '21 (2021). https://doi.org/10.1145/3442381.3449802

[15] Wu, C., Wang, C., Xu, J., Liu, Z., Zheng, K., Wang, X., Song, Y., Gai, K.: Graph contrastive learning with generative adversarial network. In: Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, pp. 2721–2730 (2023)

[16] Hemberg, E., Turner, M.J., Rutar, N., O'reilly, U.-M.: Enhancements to threat, vulnerability, and mitigation knowledge for cyber analytics, hunting, and simulations. Digital Threats **5**(1) (2024)

[17] Grigorescu, O., Nica, A., Dascalu, M., Rughinis, R.: CVE2ATT&CK: BERT-Based mapping of CVEs to MITRE ATT&CK techniques. Algorithms **15**(9) (2022)

[18] Binyamini, H., Bitton, R., Inokuchi, M., Yagyu, T., Elovici, Y., Shabtai, A.: A framework for modeling cyber attack techniques from security vulnerability descriptions. Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery &; Data Mining, 2574–2583 (2021) https://doi.org/10.1145/3447548.3467159

[19] Kuppa, A., Aouad, L., Le-Khac, N.-A.: Linking CVE's to mitre ATT&CK techniques. In: Proceedings of the 16th International Conference on Availability, Reliability and Security. ARES '21 (2021)

[20] Kanakogi, K., Washizaki, H., Fukazawa, Y., Ogata, S., Okubo, T., Kato, T., Kanuka, H., Hazeyama, A., Yoshioka, N.: Comparative evaluation of NLP-based approaches for linking CAPEC attack patterns from CVE vulnerability information. Applied Sciences (Switzerland) **12**(7) (2022)

[21] Alexander, O., Belisle, M., Steele, J.: MITRE ATT&CK for industrial control systems: Design and philosophy. The MITRE Corp.: Bedford, MA, USA **29** (2020)

[22] Xiong, W., Legrand, E., Åberg, O., Lagerström, R.: Cyber security threat modeling based on the mitre enterprise ATT&CK matrix. Software and Systems Modeling **21**(1), 157–177 (2022)

[23] Corporation, M.: Common Weakness Enumeration (CWE) Dataset. Accessed: 2024-06-02 (2024). https://cwe.mitre.org/

[24] Beerman, J., Berent, D., Falter, Z., Bhunia, S.: A review of colonial pipeline ransomware attack. In: 2023 IEEE/ACM 23rd International Symposium on Cluster, Cloud and Internet Computing Workshops (CCGridW), pp. 8–15 (2023)

[25] Velickovic, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., Bengio, Y., *et al.*: Graph attention networks. Stat **1050**, 10–48550 (2017)