Coupled-Space Attacks Against Random-Walk-Based Anomaly Detection

Yuni Lai, Marcin Waniek, Liying Li, Jingwen Wu, Yulin Zhu, Tomasz P. Michalak, Talal Rahwan, and Kai Zhou, *Member, IEEE*

***Abstract*— Random Walks-based Anomaly Detection (RWAD) is commonly used to identify anomalous patterns in various applications. An intriguing characteristic of RWAD is that the input graph can either be pre-existing graphs or feature-derived graphs constructed from raw features. Consequently, there are two potential attack surfaces against RWAD: graph-space attacks and feature-space attacks. In this paper, we explore this vulnerability by designing practical coupled-space (interdependent feature-space and graph-space) attacks, investigating the interplay between graph-space and feature-space attacks. To this end, we conduct a thorough complexity analysis, proving that attacking RWAD is NP-hard. Then, we proceed to formulate the graph-space attack as a bi-level optimization problem and propose two strategies to solve it: alternative iteration (alterI-attack) or utilizing the closed-form solution of the random walk model (cf-attack). Finally, we utilize the results from the graph-space attacks as guidance to design more powerful feature-space attacks (i.e., graph-guided attacks). Comprehensive experiments demonstrate that our proposed attacks are effective in enabling the target nodes to evade the detection from RWAD with a limited attack budget. In addition, we conduct transfer attack experiments in a black-box setting, which show that our feature attack significantly decreases the anomaly scores of target nodes. Our study opens the door to studying the coupled-space attack against graph anomaly detection in which the graph space relies on the feature space.**

***Index Terms*— Graph-based anomaly detection, random walk, poisoning attack, adversarial attacks, security and privacy.**

I. Introduction

GRAPH-BASED Anomaly Detection (GAD) has gained significant research attention in recent years due to the widespread use of graph data across various application domains. GAD algorithms are designed to identify anomalies in a graph, where nodes represent entities, and edges indicate their relations. Essentially, a GAD algorithm works by initially measuring the similarities among nodes and then identifying nodes that are less similar to the rest as anomalous. Despite the development of supervised GADs, such as GADs based on graph neural networks (GNNs) [1], unsupervised GADs still have advantages in their simplicity, unsupervised property, and effectiveness. Random Walks (RWs), such as PageRank [2], have emerged as a powerful tool for measuring node similarities over graphs and have become a fundamental component of many unsupervised GAD systems that are extensively employed in diverse applications. Notably, Random-Walk-based Anomaly Detection (RWAD) has been employed in detecting money laundering within the financial industry [3], identifying fraudsters in online shopping [4], uncovering fake accounts in social networks [5], [6], [7], [8], [9], and serving as a general unsupervised outlier detection method for bipartite graphs [10], [11] (e.g., review data in recommender systems, stock market transaction data, and short message service), multivariate time series data [12], [13] (e.g., electrocardiograms data), and the most common feature data [14], [15], [16], [17], [18] (e.g., network intrusion detection data). Moreover, random walk has also been adopted to improve large-scale graph anomalies detection [19] and enhance deep-learning-based anomalies detection [20], [21]. These diverse applications underscore the important role of RWAD in ensuring system security.

As the accuracy of predictions produced by the RWAD methods is crucial for system security, it is essential to assess their robustness in a real-world adversarial environment. In fact, the individuals that RWAD aims to detect may have both the incentive and capability to evade detection. For instance, adversaries controlling bank accounts to be used in money laundering schemes may wish to remain undetected to continue their malicious activities. They could carefully manage the everyday transactions on the accounts to make them appear similar to normal ones, causing the system to falsely classify them as benign. In essence, in an adversarial environment, attackers can intentionally manipulate the input data to RWAD in order to mislead its predictions, leading to what is known as *data poisoning attacks* in the literature. However, studying the adversarial robustness of RWAD imposes new challenges due to an intriguing characteristic of RWAD. Specifically, in an RWAD system, the graph is often not directly accessible and needs to be *constructed* from raw data. As illustrated in Fig. 1 (top), entities in the system are

Received 23 October 2023; revised 24 April 2024 and 13 August 2024; accepted 16 September 2024. Date of publication 25 September 2024; date of current version 9 October 2024. This work was supported in part by the National Science Foundation of China under Grant 62106210 and in part by the Hong Kong Research Grants Council (RGC) Project under Grant PolyU25210821. The work of Tomasz P. Michalak was supported by the European Research Council (ERC) "PROCONTRA" under Grant 885666. The associate editor coordinating the review of this article and approving it for publication was Prof. Kemal Akkaya. *(Corresponding author: Kai Zhou.)*

Yuni Lai, Liying Li, Jingwen Wu, Yulin Zhu, and Kai Zhou are with the Department of Computing, The Hong Kong Polytechnic University, Hong Kong (e-mail: kaizhou@polyu.edu.hk).

Marcin Waniek and Tomasz P. Michalak are with the Institute of Informatics, University of Warsaw, 02-093 Warszawa, Poland.

Talal Rahwan is with the Department of Computer Science, New York University Abu Dhabi, Abu Dhabi, United Arab Emirates.

Digital Object Identifier 10.1109/TIFS.2024.3468156

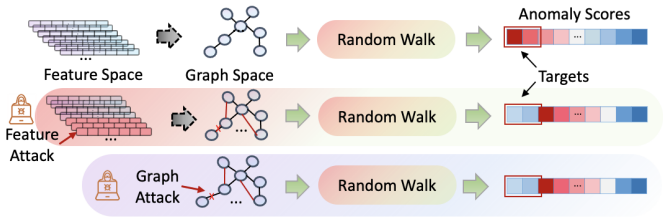


Fig. 1. Illustration of RW-based anomaly detection and the distinction between graph-space and feature-space attacks.

represented as vectors in a feature space, and a graph is then constructed based on the relationships among the entities *as determined by their feature vectors*. This kind of graph is termed as feature-derived graph. For instance, a proximity graph can be constructed based on feature similarity. This graph is then fed into the RWAD system, which produces anomaly scores for each node.

Consequently, there are *two potential attack surfaces* against RWAD: **graph-space** attacks and **feature-space** attacks. In graph-space attacks (Fig. 1, bottom), the attacker can directly modify the structure of the graph, which is a common assumption made by previous works [22], [23], [24] that design structural attacks on graphs. In feature-space attacks (Fig. 1, middle), the attacker does not have direct control over the graph but can modify the features, which indirectly affects the graph's structure. It is worth noting that in the latter case, where the graph is not directly accessible, feature-space attacks are deemed more realistic (further explained in Section VII-A).

Unfortunately, previous research treats attacks in the graph space and feature space rather separately. On the one hand, many existing works have investigated *structural attacks* [22], [24], [25], [26] against a wide range of graph learning models. On the other hand, another line of research has focused on studying *feature manipulation attacks* [27], [28], [29] primarily in the computer vision domain, where the data objects represented by features are independent of each other. In contrast, one unique characteristic of RWAD is that it examines data objects that are interdependent. Specifically, the data processing pipeline of RWAD involves transforming the features into graphs, over which the random walk operates. That is, the data in the feature space and the data in the graph space are interdependent in the sense that any modifications to the features will be reflected in the changes in the constructed graphs. This unique interdependency makes the interplay between the graph-space and feature-space attacks possible.

Thus, for the first time, we aim to investigate the adversarial robustness of RWAD under ***coupled-space*** attacks, where the attackers can explicitly exploit the interdependency between two coupled data spaces to effectively achieve their malicious goals. Our main motivations for exploring coupled-space attacks are twofold. First, data manipulation in the feature space is more realistic since the graph is constructed *virtually* in the pipeline of which the attacker does not have direct control. Second, since random walks directly run over the constructed graphs, an attacker can potentially leverage the anticipated data manipulation in the graph space to guide the modifications in the feature space, which can make the attack more effective.

Towards this end, we begin with a formal analysis of graph-space attacks. The simplicity of RWAD allows us to conduct a hardness analysis. Specifically, we define the attacks in the graph space as a decision problem. We ask whether an attacker can reduce the anomaly scores of the target nodes below a certain threshold, thereby classifying them as benign, by modifying a limited number of edges in a given graph. Our in-depth complexity analysis shows that this problem is NP-hard for both *directed* and *undirected* graphs. Furthermore, since feature-space attacks ultimately modify edges, they can be viewed as special cases of this problem, and the hardness results remain applicable. The hardness results serve as the anchor for us to investigate efficient attack algorithms in both the graph space and feature space.

We then proceed to design effective graph-space attacks, which are formulated as an optimization problem with the objective of minimizing the target nodes' anomaly scores output by RWAD. Solving this optimization problem encounters several challenges. Firstly, random walk (PageRank) is an iterative algorithm that operates on an input graph; thus, any changes made to the graph will require the iterations to be re-executed. Consequently, attacks against RWAD will result in a *bi-level* optimization where the inner layer involves complex iteration. Second, the discrete nature of graph structure further complicates the solving of the optimization. To address these challenges, we propose two efficient attacks: **alterI**-attack and **cf**-attack. The former is an iterative approach that optimizes the attack objective by projected gradient descent (PGD) [30] and updates the random walk model alternatively. The latter utilizes the closed form of the random walk model to transform the bi-level optimization into a single-level problem.

Finally, we investigate the more realistic feature-space attacks. Our major innovation is to use the results from the *virtual* graph-space attack as our guidance to design more powerful feature-space attacks. Specifically, we utilize the guidance from two aspects: selecting the attack nodes and formulating an effective attack objective. Through extensive experiments, we demonstrate that by fully exploring the dynamics between attacks in coupled spaces, more powerful attacks could be designed, revealing more realistic security threats against RWAD systems.

The main contributions are summarized as follows:

- We study the adversarial robustness of RWAD, for the first time, exploring the interplay between attacks in coupled spaces.
- We present a deep theoretical analysis of the hardness of attacking RWAD, which is proved to be NP-hard on both directed and undirected graphs.
- We propose effective attacks in coupled spaces. In particular, we innovatively utilize the results from the graph-space attacks as guidance to design more powerful feature-space attacks.
- We conduct comprehensive experiments to demonstrate the effectiveness of our proposed attacks. Especially we also transfer our attacks to other anomaly detection methods in the feature space. It is shown that our graph-guided

feature-space attack remains effective even without knowing the target models, demonstrating a realistic threat in real-world application scenarios.

In summary, our work uncovers a unique vulnerability of RWAD and unleashes the power of attackers by exploring the interplay between attacks in coupled spaces, significantly advancing our knowledge of the adversarial robustness of RWAD in deployment.

Road Map: Related works (II) ⇒ Target RWAD models (III) ⇒ Problem statements (IV) ⇒ Complexity analysis of attacks (V) ⇒ Effective graph-space attacks (VI) ⇒ Graph-guided feature-space attacks (VII) ⇒ Evaluation (VIII) ⇒ Limitation and Future Work (IX) ⇒ Conclusion (X).

II. Related Works

A. Graph-Based Anomaly Detection

This paper focus on unsupervised and node-level anomaly detection on plain and static graph. *Random-walk-based techniques* [10], [12], [14], [15], [16], [17], [31] discussed in this paper, exploiting random walk as a similarity or connectivity measurement. Traditional *feature-based* techniques [32], [33], [34] utilize statistical features, such as in and out node degrees, to extract structural information from graphs and transform the GAD to usual anomaly detection problem. For example, OddBall [32] built a regression model based on the density power law to estimate anomalous local patterns. These labor-intensive handcrafted features have limitations on generalizing to unknown anomalies. Beyond handcrafted features, *network-representation-based* techniques, such as DeepWalk [35] and Node2Vec [36], are widely exploited to extract a more flexible feature representation which can be used for downstream anomaly detection tasks [37]. Most recent work mainly focuses on investigating *deep learning based* anomaly detection, such as DOMINANT [38], GAL [20], TAM [39], GLAD [40], and GAD-NR [41].

B. Adversarial Attacks on Graph

Our work belongs to the category of targeted and poisoning adversarial attacks. Here, we include the most related existing attacks on graphs. There are some previous research efforts on the random walk (RW) based models. Reference [42] reformulate the DeepWalk model as a matrix factorization form to reduce the bi-level optimization to single-level, and then optimize the untargeted attack loss by optimizing the graph spectrum. Reference [43] make further improvement to make the spectrum-based attack works in a black-box system. Different from our attacks on RW-based anomaly detection, they mainly focus on attacking node embedding generated by RW.

In addition to RW-based model, Nettack [22], Metattack [23] are two strong poisoning attacks for the GCN-based models. Nettack greedily selects the perturbation edges among the candidate sets with the largest gradient obtained by incremental updates. Metattack greedily selects the perturbation edges with the largest gradient obtained by meta-gradient. Note that, both of these methods can be extended to attack node features. However, Nettack does not introduce the attack node selection, and Metattack is only applicable to binary features. Furthermore, the proximity graph is different from other graphs. The proximity graph is changing along with features, while the node feature attack in [22] and [23] have fixed graph structures. For belief propagation models, [30] introduced a poisoning attack for graph data. For another classical graph-based anomaly detection model called OddBall, [24] proposed BinarizedAttack which is well-designed for the binary property of edges. For graph contrastive learning, [44] attack the graph embedding by greedily choosing the most informative edges. Beyond gradient-based methods, perturbing the intrinsic property of graphs, such as spectral changes [45] shows to be more effective, but it is only suitable for untargeted attacks. These works are orthogonal to our study.

III. Random-Walk-Based Anomaly Detection

In this section, we introduce the necessary background on unsupervised random-walk-based anomaly detection (RWAD). We first present an overview of the framework with an emphasis on the role of random walk (RW) in anomaly detection, and then give two concrete exemplar RWAD models, which are also the target models considered in this paper.

A. Overview

1) Input Data as a Graph: In general, RWAD takes a *plain* graph as input and produces anomaly scores for the nodes in the graph as output. In practice, the input graph could be either directly available or constructed from raw data. Depending on the levels of accessibility of the graph, we divide RWAD systems into two types:

- RWAD over *directly accessible graph* (*Di-RWAD*): In this case, the input to RWAD is a graph that represents relational data in a specific application. For instance, in recommender systems, the rating towards products given by customers on E-commerce platforms can be modeled as a *bipartite graph*.
- RWAD over *indirectly accessible graph* (*InDi-RWAD*): In this case, the input to RWAD are raw features of entities, and a graph is constructed as a data preprocessing step in the pipeline of anomaly detection (Fig. 1, top). Typically, given the feature vectors, a *proximity graph* is constructed, where the nodes represent entities and an edge exists between two nodes only if they are similar enough in certain similarity metrics [14], [15], [16], [17].

We note that in both cases, RWAD will operate on graphs; however, the difference lies in whether the graph is directly accessible. Later we will see that such a difference is crucial for determining the attacker's ability when designing attacks.

2) RW as a Similarity Measurement: The core of unsupervised anomaly detection is to identify data points that are significantly different from the rest of the population. RW has been shown to be an effective method for measuring the similarities of nodes in a graph. Specifically, given a graph $G = (V, E)$ with its adjacency matrix denoted as W, we define the transition matrix $P = (p_{ij})_{|V|\times|V|}$ as the column-normalized version of the adjacency matrix W, where $p_{ij} = w_{ij} / \sum_{t=1}^{|V|} w_{i,t}$. If vertex i has no outgoing edges

(i.e., $\sum_{t=1}^{k+n} w_{i,t} = 0$), we set the transition probability to 0. The widely used Page-Rank algorithm with restart can be represented as follows:

$$\vec{s} = (1-\alpha)P\vec{s} + \alpha\vec{r}, \tag{1}$$

where α is the restart rate, a hyper-parameter that controls the probability of restart; the vector $\vec{r}$ specifies the restart strategy, and $\vec{s}$ characterizes the node similarities. With the similarity, the anomaly score of a node is calculated as the opposite of its average similarity to all other nodes, or the average similarity among its neighbors. Next, we present two representative models to instantiate the *Di-RWAD* and *InDi-RWAD* systems.

B. Representative Target Models

1) Di-RWAD: We consider bipartite graphs as a representative example of directly accessible graphs. We next describe how to apply the RWAD algorithm to the bipartite graphs of this kind, which we term as *BiGraphRW* model.

To begin, we define a bipartite graph $G = (U \cup V, E)$ as a graph with two disjoint sets of vertices $U = \{u_i | 1 \le i \le k\}$ and $V = \{v_i | 1 \le i \le n\}$, and a set of edges $E \subseteq U \times V$ that connect the vertices in U to the vertices in V. We represent the edges in E as a binary edge matrix $M = (m_{ij})k \times n$, where $m_{ij} = 1$ if $\langle i, j\rangle \in E$, and $m_{ij} = 0$ otherwise. Then, the adjacency matrix for a bipartite graph can be constructed as $W = (w_{ij})_{(k+n)\times(k+n)} = \begin{pmatrix} 0 & M \\ M^T & 0 \end{pmatrix}$.

For each node $u \in U$, *BiGraphRW* applies Eqn. 1 with $\vec{r} = \vec{e}_u$, where $\vec{e}_u$ is a vector with zeros element except node u, which means that it always restarts from node u. The resulting vector $\vec{s}_u = (1-\alpha)P\vec{s}_u + \alpha\vec{e}_u$ represents the connectivity scores of node pairs $\{\langle u, t\rangle | t \in U \cup V\}$, which quantifies the similarity between node u and others. By assumption, a node v tends to have a lower mean similarity score among its neighbors if it is anomalous. We denote the average neighbor similarity as $\bar{S}_v$:

$$\bar{S}_v = \frac{\sum_{i=1}^{k} M_{iv} \sum_{j=1, i\neq j}^{k} M_{jv} \vec{s}_{u_i}(u_j)}{\sum_{i=1}^{k} M_{iv} \sum_{j=1, i\neq j}^{k} M_{jv}}, \tag{2}$$

where $\vec{s}_{u_i}(u_j)$ represent the element corresponding to node u_j in $\vec{s_{u_i}}$, which is the similarity between node u_i and u_j. Anomaly score of node v is in contract to the mean similarity score $\bar{S}_v$, so we denoted it by

$$\mathcal{A}(v) = 1 - \bar{S}_v = \begin{cases} \text{anomaly}, & \text{if } \mathcal{A}(v) \ge \theta, \\ \text{normal node}, & \text{if } \mathcal{A}(v) < \theta, \end{cases} \tag{3}$$

where the parameter θ is a given and fixed threshold of the anomaly detection model.

2) InDi-RWAD: A representative way to apply RWAD to non-graph data is by constructing a proximity graph. We call this variant as *ProxGraphRW* model. In this approach, the input feature data is represented as $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n]$, $\mathbf{x}_i \in \mathbb{R}^d$. The first step is to construct a proximity graph according to the similarity or distance measurement between each pair of samples. To construct a proximity graph $G = (V, E)$, the vertices V represent data samples $\{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n\}$, and the edges imply the similarity among vertices. This can be achieved through similarity measures, such as Euclidean distance, cosine similarity, or correlation coefficient. We denote the similarity function between $\mathbf{x}_i$ and $\mathbf{x}_j$ as $sim(\mathbf{x}_i, \mathbf{x}_j)$. Then, proximity graphs can be constructed by different rules. In this paper, we take ϵ-Graph [14], [15] as an example, where for every data sample $\mathbf{x}_i$, an edge is connected to $\mathbf{x}_j$ if $sim\left(\mathbf{x}_i, \mathbf{x}_j\right) > \epsilon$. We define the weighted adjacency matrix as $W = (w_{ij})_{n\times n}$, where $w_{ij} = sim(\mathbf{x}_i, \mathbf{x}_j) \cdot \mathbb{I}(sim(\mathbf{x}_i, \mathbf{x}_j) > \epsilon)$, and $\mathbb{I}(\cdot)$ is an indicator function. With the proximity graph constructed, *ProxGraphRW* applies the Eqn. 1 with $\vec{r} = \frac{1}{n}$, which means that the RW restart from any node with equal probability. The resulting vector $\vec{s} = (1-\alpha)P\vec{s} + \frac{\alpha}{n}$ contains the connectivity scores of all nodes, where each element $\vec{s}(v)$ quantifies the overall similarity of node v to all other nodes. Finally, based on the hypothesis that anomalies have low connectivity to most others, the anomaly score of node v is

$$\mathcal{A}(v) = 1 - \vec{s}(v), \tag{4}$$

where $\vec{s}(v)$ is the element corresponding to node v in $\vec{s}$.

IV. Problem Statements

In this section, we introduce the adversarial environment that random-walk-based anomaly detection (RWAD) operates in, and then formally define the attack problem.

A. System and Threat Model

We consider a system consisting of two parties: an analyst who runs an RWAD algorithm to detect potential anomalies and an attacker who aims to evade the detection. In practice, the analyst would first collect data from the environment and construct a graph, which is fed into the RWAD system for anomaly detection. However, the attacker could tamper with the data collection process which will result in a poisoned graph, leading to the malfunction of the system. For instance, in online shopping platforms, the attacker may manipulate some users to provide fake ratings for target items. The resulting poisoned data can lead to biased recommendations from the recommender system.

We further introduce the threat model by specifying the attacker's knowledge, goal, and capability. By Kerckhoffs's principle, we assume a worst-case scenario where the attacker knows all the data as well as the anomaly detection model, which is a common assumption employed by many previous attacks [24], [46]. We assume that the attacker has a set of target nodes in mind. Initially, the target nodes would have been determined as abnormal by the RWAD system if no data was manipulated. The attacker then tries to decrease the anomaly scores of those target nodes in the hope that they would evade the detection. To this end, the attacker can manipulate the data constrained by a certain budget. Specifically, depending on whether the graph is directly accessible or not, we divide the attacks into two types:

- *Graph-space* attack: the attacker can directly modify the structure of the graph by adding and deleting the edges under a budget constraint K.

TABLE I
HARDNESS RESULTS OF **PA-RWAD**

	Directed graph	Undirected graph
PA-RWAD	NP-hard (**Lemma 1** & **Theom. 1**)	NP-hard (**Theom. 2**)

- *Feature-space* attack: the attacker can only modify the features of a set of attack nodes, which will indirectly cause changes in the graph structure. Considering a practical scenario that the targeted anomaly nodes are crafted to have specific malicious functions, we can not modify their features arbitrarily. Therefore, an indirect feature attack, aiming to decrease the anomaly scores of target nodes while keeping their features unchanged, is ideal for such a problem. Hence, we restrict the selection of attack nodes to those other than the target nodes.

B. Problem Definition

To facilitate our theoretical analysis, we formally define the attacks against RWAD as follows.

*Definition 1 (**PA-RWAD**: poisoning attacks against RWAD): An instance of the problem is defined by a tuple, $(G, \mathcal{T}, \mathcal{A}, \Theta, K, \widehat{A}, \widehat{R})$, where $G = (V, E)$ is a network, $\mathcal{T} \subseteq V$ is the set of targets, $\mathcal{A} : \mathbb{G} \times V \to \mathbb{R}$ is the anomaly score function, $\Theta \in \mathbb{N}$ is the safety threshold, $K \in \mathbb{N}$ is the budget specifying the maximum number of edges that can be added or removed, $\widehat{A} \subseteq (V \times V) \setminus E$ is the set of edges that can be added, and $\widehat{R} \subseteq E$ is the set of edges that can be removed. The goal is then to identify two sets, $A^* \subseteq \widehat{A}$ and $R^* \subseteq \widehat{R}$, such that $|A^*| + |R^*| \leq K$, and for $G^* = (V, (E \cup A^*) \setminus R^*)$ we have:*

$$\left|\left\{v_i \in V : \forall_{v_j \in \mathcal{T}} \mathcal{A}(G^*, v_i) > \mathcal{A}(G^*, v_j)\right\}\right| \geq \Theta.$$

In practice, the top-Θ nodes ranked by their anomaly scores in descending order are determined as anomalous. Then, the goal of **PA-RWAD** is to find a way of modifying the network by adding and removing edges, so that there are at least Θ nodes with anomaly scores greater than any of the target nodes. In other words, the target nodes are considered as benign.

We note that although **PA-RWAD** emphasizes modifying the structure of the graph, a feature-space attack is still an instance of **PA-RWAD**, since the modification of features will ultimately lead to the changes of the graph.

V. COMPLEXITY ANALYSIS

We now proceed to analyze the computational complexity of the attacks against RWAD. We summarize the hardness results in Tab. I.

*Theorem 1: The **PA-RWAD** problem is NP-hard given a directed graph.*

Proof: We will prove that the problem is NP-hard by showing a reduction from the NP-complete *3-Set Cover* problem. An instance of this problem is defined by a collection of subsets $Q = \{Q_1, \ldots, Q_{|Q|}\}$ of the universe $U = \{u_1, \ldots, u_{|U|}\} = \bigcup_{Q_i \in Q} Q_i$ such that $\forall |Q_i| = 3$, and a number $k \in \mathbb{N}$. The goal is to determine whether there exist at most k elements of Q that cover the entire universe, i.e., $Q^* \subseteq Q$ such that $|Q^*| \leq k$ and $U = \bigcup_{Q_i \in Q^*} Q_i$.

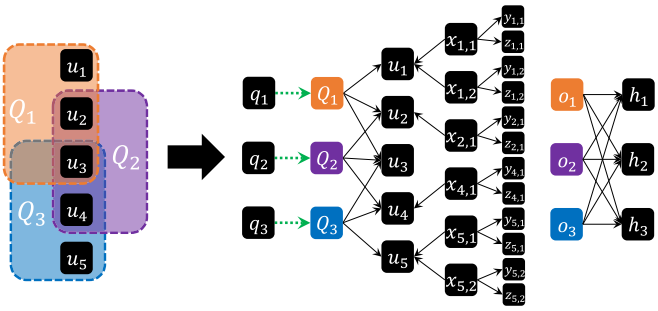


Fig. 2. An example of the construction used in the proof of Theorem 1. The green dotted arrows represent edges that can be added.

Let (Q, k) be a given instance of the 3-Set Cover problem. We will now construct an instance of the **PA-RWAD** problem. In what follows, let $Q(u_i)$ be the subsets in Q that contain u_i, i.e., $Q(u_i) = \{Q_j \in Q : u_i \in Q_j\}$. Let us also assume that $|Q| \geq 4$, as all smaller instances can be easily solved in constant time. First, we construct a directed network $G_Q = (V, E)$, where:

- $V = U \cup \bigcup_{Q_i \in Q} \{Q_i, q_i, o_i\} \cup \{h_1, h_2, h_3\} \cup \bigcup_{u_i \in U} \bigcup_{j=1}^{|Q|-|Q(u_i)|} \{x_{i,j}, y_{i,j}, z_{i,j}\}$,
- $E = \bigcup_{u_i \in Q_j} \{(Q_j, u_i)\} \cup \bigcup_{o_i \in V} \bigcup_{h_j \in V} \{(o_i, h_j)\} \cup \bigcup_{x_{i,j} \in V} \{(x_{i,j}, u_i), (x_{i,j}, y_{i,j}), (x_{i,j}, z_{i,j})\}$.

An example of this construction (e.g., $|U| = 5$, $|Q| = 3$) is presented in Fig. 2. Now, consider the instance $(G_Q, \mathcal{T}, \mathcal{A}, \Theta, K, \widehat{A}, \widehat{R})$ of the **PA-RWAD** problem, where:

- G_Q is the network we just constructed,
- $\mathcal{T} = U$ is the target set,
- $\mathcal{A}$ is the anomaly score function with the restart rate parameter $\alpha = \frac{1}{|Q|}$,
- $\Theta = n - |U|$ is the safety threshold,
- $K = k$ is the budget,
- $\widehat{A} = \bigcup_{Q_i \in Q} \{(q_i, Q_i)\}$, i.e., only edges from q_i to corresponding Q_i can be added,
- $\widehat{R} = \emptyset$, i.e., none of the edges can be removed.

Since $\widehat{R} = \emptyset$, for any solution to the constructed instance of the **PA-RWAD** problem, we must have $R^* = \emptyset$. Hence, we will omit the mentions of R^* in the remainder of the proof, and we will assume that a solution consists just of A^*. We next prove a useful lemma.

Lemma 1: Let $A \subseteq \bigcup_{Q_i \in Q} \{(q_i, Q_i)\}$, and let $G_Q \cup A = (V, E \cup A)$. We have that:

$$\forall_{u_i \in U} \forall_{v \notin U} \mathcal{A}(G_Q \cup A, v) > \mathcal{A}(G_Q \cup A, u_i)$$

if and only if $\forall_{u_i \in U} \exists_{(q_j, Q_j) \in A} u_i \in Q_j$.

Proof: From the formula of the anomaly score function, we have that $\mathcal{A}(G_Q \cup A, v_i) = 1 - \vec{s}(G_Q \cup A, v_i)$, where:

$$\vec{s}(G_Q \cup A, v_i) = \frac{\alpha}{n} + (1 - \alpha) \sum_{v_j \in V} \vec{s}(G_Q \cup A, v_j) P_{j,i}.$$

Therefore, we have that $\mathcal{A}(G_Q \cup A, v_i) > \mathcal{A}(G_Q \cup A, v_j)$ if and only if $\vec{s}(G_Q \cup A, v_i) < \vec{s}(G_Q \cup A, v_j)$. Let $A(u_i)$ be the set of Q_j containing u_i that got connected to the corresponding node q_j via the edges in A, i.e., $A(u_i) = \{Q_j \in Q : u_i \in Q_j \wedge$

$(q_j, Q_j) \in A\}$. We now compute the values of $\vec{s}(G_Q \cup A, v_i)$ for all nodes in V:

- $\vec{s}(G_Q \cup A, q_i) = \vec{s}(G_Q \cup A, x_{i,j}) = \vec{s}(G_Q \cup A, o_i) = \frac{\alpha}{n} = \frac{1}{|Q|n}$, as nodes q_i, $x_{i,j}$, and o_i do not have any predecessors,
- $\vec{s}(G_Q \cup A, y_{i,j}) = \vec{s}(G_Q \cup A, z_{i,j}) = \frac{\alpha}{n} + (1-\alpha)\vec{s}(G_Q \cup A, x_{i,j})\frac{1}{3} = \frac{\alpha}{n} + \frac{(1-\alpha)\alpha}{3n} = \frac{(4-\alpha)\alpha}{3n} = \frac{\left(4-\frac{1}{|Q|}\right)}{3|Q|n}$, as the only predecessor of nodes $y_{i,j}$ and $z_{i,j}$ is the node $x_{i,j}$ with out-degree 3,
- $\vec{s}(G_Q \cup A, h_i) = \frac{\alpha}{n} + (1-\alpha)\sum_{o_j \in V} \vec{s}(G_Q \cup A, o_j)\frac{1}{3} = \frac{\alpha}{n} + \frac{|Q|(1-\alpha)\alpha}{3n} = \frac{((1-\alpha)|Q|+3)\alpha}{3n} = \frac{|Q|+2}{3|Q|n}$, as the predecessors of h_i are all $|Q|$ nodes o_j, each with out-degree 3,
- if $(q_i, Q_i) \notin A$ then $\vec{s}(G_Q \cup A, Q_i) = \frac{\alpha}{n} = \frac{1}{|Q|n}$, as such node Q_i has no predecessors,
- if $(q_i, Q_i) \in A$ then $\vec{s}(G_Q \cup A, Q_i) = \frac{\alpha}{n} + (1-\alpha)\vec{s}(G_Q \cup A, q_i) = \frac{\alpha}{n} + (1-\alpha)\frac{\alpha}{n} = \frac{2|Q|-1}{|Q|^2 n}$, as the only predecessor of such node Q_i is the node q_i,
- $\vec{s}(G_Q \cup A, u_i) = \frac{\alpha}{n} + (1-\alpha)\sum_{Q_j \in Q(u_i)} \vec{s}(G, Q_j)\frac{1}{3} + (1-\alpha)\sum_{x_{i,j} \in V} \vec{s}(G, x_{i,j})\frac{1}{3} = \frac{\alpha}{n} + \frac{|Q|(1-\alpha)\alpha}{3n} + |A(u_i)|\frac{(1-\alpha)^2\alpha}{3n} = \frac{((1-\alpha)|Q|+3+|A(u_i)|(1-\alpha)^2)\alpha}{3n} = \frac{|Q|+2+|A(u_i)|\left(1-\frac{1}{|Q|}\right)^2}{3|Q|n}$, as the predecessors of u_i are $|Q(u_i)|$ nodes Q_j, as well as $|Q| - |Q(u_i)|$ nodes $x_{i,j}$, each with out-degree 3.

We now prove the main equivalence of the lemma. Assume that $\forall_{u_i \in U} \forall_{v \notin U} \mathcal{A}(G_Q \cup A, v) > \mathcal{A}(G_Q \cup A, u_i)$. In particular, it implies that: $\forall_{u_i \in U}\, \vec{s}(G_Q \cup A, u_i) - \vec{s}(G_Q \cup A, h_1) > 0$. By substituting the values in the inequality, we get:

$$\forall_{u_i \in U} \frac{|A(u_i)|\left(1-\frac{1}{|Q|}\right)^2}{3|Q|n} > 0,$$

which in turn implies that $\forall_{u_i \in U} |A(u_i)| > 0$. Hence, we have that for every $u_i \in U$ there exists at least one Q_j such that $u_i \in Q_j$ and $(q_j, Q_j) \in A$.

To prove the implication in the other direction, assume that $\forall_{u_i \in U}\, \exists_{(q_j, Q_j) \in A} u_i \in Q_j$. Hence, we get that $\forall_{u_i \in U} |A(u_i)| > 0$, which implies that: $\forall_{u_i \in U}\, \vec{s}(G_Q \cup A, u_i) \geq \frac{|Q|+2+\left(1-\frac{1}{|Q|}\right)^2}{3|Q|n}$. By comparing this value to the values computed above, we have that $\forall_{u_i \in U}, \forall_{v \notin U}$:

$$\vec{s}(G_Q \cup A, v) < \frac{|Q|+2+\left(1-\frac{1}{|Q|}\right)^2}{3|Q|n} \leq \vec{s}(G_Q \cup A, u_i),$$

which in turn implies that:

$$\forall_{u_i \in U} \forall_{v \notin U}\, \mathcal{A}(G_Q \cup A, v) > \mathcal{A}(G_Q \cup A, u_i).$$

This concludes the proof of the lemma. □

Let $Q^* \subseteq Q$ be a solution to the given instance of the 3-Set Cover problem, i.e., $|Q^*| \leq k$ and $\forall_{u_i \in U} \exists_{Q_j \in Q^*} u_i \in Q_j$. From Lemma 1 we have that $\mathcal{A}(G_Q \cup A^*, v) > \mathcal{A}(G_Q \cup A^*, u_i)$ where $A^* = \{(q_i, Q_i) : Q_i \in Q^*\}$. Hence, in network $G_Q \cup A^*$ all $\Theta = n - |U|$ nodes other than the nodes in U have greater anomaly scores than all the nodes in U, and $|A^*| \leq k = K$.

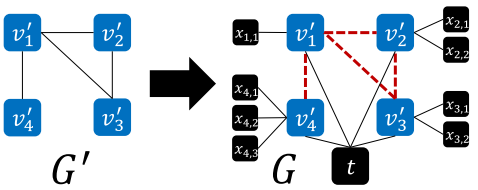


Fig. 3. An example of the construction used in the proof of Theorem 2. The red dashed lines represent edges that can be removed.

Therefore, A^* is a solution to the constructed instance of the **PA-RWAD** problem.

To prove the implication in the other direction, assume that A^* is a solution to the constructed instance of the **PA-RWAD** problem. In particular, it implies that $|A^*| \leq K = k$ and $\forall_{u_i \in U} \forall_{v \notin U} \mathcal{A}(G_Q \cup A, v) > \mathcal{A}(G_Q \cup A, u_i)$. From Lemma 1 we have that $\forall_{u_i \in U} \exists_{(q_j, Q_j) \in A} u_i \in Q_j$. Therefore, $\{Q_i \in Q : (q_i, Q_i) \in A^*\}$ is a solution to the given instance of the 3-Set Cover problem.

We have shown that the constructed instance of the **PA-RWAD** problem has a solution if and only if the given instance of the 3-Set Cover problem has a solution, which concludes the proof of NP-hardness. □

*Theorem 2: The **PA-RWAD** problem is NP-hard given an undirected graph.*

Proof: We will prove that the problem is NP-hard by showing a reduction from the NP-complete *Finding k-Clique* problem. An instance of this problem is defined by a network $G' = (V', E')$, and a number $k \in \mathbb{N}$. The goal is to determine whether there exist k nodes that induce a clique in G'.

Let (G', k) be a given instance of the Finding k-Clique problem. We will now construct an instance of the **PA-RWAD** problem. Let $n' = |V'|$, and let $d(G, v)$ be the degree of v in network G, i.e., $d(G, v) = |\{w \in V : (v, w) \in E\}|$. First, we construct a undirected network $G = (V, E)$, where:

- $V = V' \cup \{t\} \cup \bigcup_{v'_i \in V'} \bigcup_{j=1}^{n'+k-d(G', v')-3} \{x_{i,j}\}$,
- $E = E' \cup \bigcup_{v'_i \in V'} \{(t, v'_i)\} \cup \bigcup_{x_{i,j} \in V} \{(v'_i, x_{i,j})\}$.

An example of this (e.g., $|V'| = 4, k = 3$) construction is presented in Fig. 3. Now, consider the instance $(G, \mathcal{T}, \mathcal{A}, \Theta, K, \widehat{A}, \widehat{R})$ of the **PA-RWAD** problem, where:

- G is the network we just constructed,
- $\mathcal{T} = \{t\}$ is the target set,
- $\mathcal{A}$ is the anomaly score function with the restart rate parameter $\alpha = 0$,
- $\Theta = n - (n' - k + 1)$ is the safety threshold,
- $K = \frac{k(k-1)}{2}$ is the budget,
- $\widehat{A} = \emptyset$, i.e., none of the edges can be added,
- $\widehat{R} = E'$, i.e., only edges existing in G' can be removed from G.

Since $\widehat{A} = \emptyset$, for any solution to the constructed instance of the **PA-RWAD** problem, we must have $A^* = \emptyset$. Hence, we will omit the mentions of A^* in the remainder of the proof, and we will assume that a solution consists just of R^*.

From the formula of the anomaly score function with $\alpha = 0$ we have that $\mathcal{A}(G, v_i) = 1 - \vec{s}(G, v_i)$, where:

$$\vec{s}(G, v_i) = \sum_{v_j \in V} \vec{s}(G, v_j) P_{j,i}.$$

Therefore, we have that $\mathcal{A}(G, v_i) > \mathcal{A}(G, v_j)$ if and only if $\vec{s}(G, v_i) < \vec{s}(G, v_j)$.

Moreover, from Perra and Fortunato [47], we have that for the stationary distribution $\vec{s}$ of this form (i.e., for $\alpha = 0$) in an undirected network G we have that $\vec{s}(G, v_i) \sim d(G, v_i)$, i.e., the value of the entry in $\vec{s}$ for a given node is proportional to its degree. Therefore, we have that $\mathcal{A}(G, v_i) > \mathcal{A}(G, v_j)$ if and only if $d(G, v_i) < d(G, v_j)$. Let us now compute the values of $d(G, v_i)$ for all nodes in G:

- $d(G, t) = n'$, as the node t is connected with all n' nodes v'_i,
- $d(G, x_{i,j}) = 1 < d(G, t)$, as each node $x_{i,j}$ is only connected with the node v'_i,
- $d(G, v'_i) = 1 + d(G', v'_i) + n' + k - d(G', v'_i) - 3 = n' + k - 2 \geq d(G, t)$, as each node v'_i is connected with the node t, $d(G', v'_i)$ nodes from V', as well as $n' + k - d(G', v'_i) - 3$ nodes $x_{i,j}$.

Since $\Theta = n - (n' - k + 1)$, all nodes $x_{i,j}$ have a smaller degree than t, and the total number of $x_{i,j}$ is $n - n' - 1$, we need at least k out of n' nodes in V' to have a smaller degree than t in order for the safety threshold to be satisfied. However, they all have equal or greater degrees than t. Hence, the safety threshold is not satisfied in G.

Since the removal of edges from $\widehat{R}$ can only change the degrees of nodes in V', we need to decrease the degree of k of these nodes to a value smaller than that of t. For each of these k nodes we have to remove at least Δ edges incident with it, where:

$$\Delta = d(G', t) - d(G', v'_i) + 1 = n' + k - 2 - n' + 1 = k - 1.$$

Let $V^* \subseteq V'$ be a solution to the given instance of the Finding k-Clique problem, i.e., a set of k nodes forming a clique in G'. Since $\widehat{R} = E'$ and the degree of each node in k-clique is $k - 1$, we have that $V^* \times V^* \subseteq \widehat{R}$, and removing $V^* \times V^*$ from G decreases the degree of k nodes from V' by $\Delta = k - 1$ each. Therefore, $V^* \times V^*$ is a solution to the constructed instance of the **PA-RWAD** problem.

To prove the implication in the other direction, assume that R^* is a solution to the constructed instance of the **PA-RWAD** problem. At least $\frac{k\Delta}{2} = \frac{k(k-1)}{2}$ of the removed edges have to be incident with the k nodes from V' contributing to the safety threshold. However, since the total budget is $K = \frac{k(k-1)}{2}$, all of the removed edges have to be incident with the k nodes from V' contributing to the safety threshold, and $\frac{k(k-1)}{2}$ edges incident with k nodes constitute a clique. Since we have that $\widehat{R} = E'$, the same edges constitute a k-clique in G'. Therefore, $\bigcup_{(v'_i, v'_j) \in R^*} \{v'_i, v'_j\}$ is a solution to the given instance of the Finding k-Clique problem.

We have shown that the constructed instance of the **PA-RWAD** problem has a solution if and only if the given instance of the Finding k-Clique problem has a solution, which concludes the proof of NP-hardness. □

VI. Practical Graph-Space Attacks

In this section, we investigate practical attacks in the graph space. We note that the graph-space attack itself is important in the case where the graph is directly accessible. Moreover, as we will show later, the results of graph-space attacks provide insightful guidance for feature-space attacks.

A. Attack Formulation

We begin by formulating the decision problem **PA-RWAD** as an optimization problem. We use $G = (V, E)$ with its corresponding adjacency matrix W to represent the original clean graph. We assume that the anomaly detection system predicts node v as an anomaly if the anomaly score $\mathcal{A}(v)$ is greater than a threshold θ. The attacker aims to decrease the number of nodes in a given target set $\mathcal{T} \subset V$ that are identified as anomalies by modifying at most K edges in the graph. To represent the edge manipulations, we denote the modification by a binary matrix $B = (b_{uv})_{(|V| \times |V|)}$, where the element $b_{uv} \in \{0, 1\}$. If $b_{uv} = 0$, the edge $\langle u, v \rangle$ remains unchanged, and $b_{uv} = 1$ lead to add/delete of edge $\langle u, v \rangle$. Then the attack graph can be represented by $|W - B|$. In this paper, we consider undirected graphs where the adjacency matrix is always symmetric, and the budget constraint can be represented as $\sum_{u>v} b_{uv} \leq K$. Then the graph-space attack problem can be formulated as follows:

$$\begin{aligned} \min_{B} \quad & \sum_{v \in \mathcal{T}} \mathbb{I}(\mathcal{A}(v) > \theta), \\ s.t. \quad & b_{uv} \in \{0, 1\}, \sum_{u>v} b_{uv} \leq K, \end{aligned} \tag{5}$$

where $\mathbb{I}(\cdot)$ is a indicator function, $\mathbb{I}(\mathcal{A}(v) > \theta) = 1$ if the anomaly scores of node v is greater than θ.

B. Attack Method

To address the non-differentiable issue of the binary values in B, we adopt a relaxation strategy by representing b_{uv} in a continuous space that ranges from 0 to 1. This is denoted as $\tilde{B}$, which is subsequently converted back to binary form $\bar{B}$ after solving the optimization problem. To handle the discrete objective function in Eqn. 5, we replace it with the sum of anomaly scores among target nodes, $\mathcal{L}_a(\tilde{B}) = \sum_{v \in \mathcal{T}} \mathcal{A}(v)$, then we can re-formulate the attack problem as:

$$\begin{aligned} \min_{\tilde{B}} \quad & \mathcal{L}_a(\tilde{B}) = \sum_{v \in \mathcal{T}} \mathcal{A}(v), \\ \text{s.t.} \quad & \tilde{b}_{uv} \in [0, 1], \sum_{u>v} \bar{b}_{uv} \leq K, \end{aligned} \tag{6}$$

where $\tilde{B}$ is the relaxed and continuous adjacency matrix, $\bar{B} = (\bar{b}_{ij})$ is the discrete version of $\tilde{B}$.

To solve the challenging bi-level optimization problem, we propose two strategies: alternative iteration attack (**alterI**-attack) and closed-form attack (**cf**-attack). In brief, the **alterI**-attack iterates the inner RW model and the attack optimization alternatively to approximate the bi-level optimization, while the **cf**-attack transforms the bi-level optimization into a single-level problem. We first introduce the **alterI**-attack and then highlight the difference in the **cf**-attack.

1) alterI-Attack: The optimization of problem (6) remains a challenging task due to the need to reverse the continuous variable $\tilde{B}$ to binary $\bar{B}$ while satisfying the budget constraint. To overcome this difficulty, we first use projected gradient descent (PGD) to efficiently optimize $\tilde{B}$ without considering the budget constraint $\sum_{u>v} \bar{b}_{uv} \leq K$. Instead, we add l_2-norm regularization on the variable $\tilde{B}$: $\mathcal{L}_a(\tilde{B}) = \sum_{v\in\mathcal{T}} \mathcal{A}(v) + \lambda||\tilde{B}||_2$, where the λ is regularization coefficient. Then, we obtain the binary matrix $\bar{B}$ by selecting the top-K elements from $\tilde{B}$. This approach allows us to efficiently approximate the constrained optimization problem while ensuring that the attack budget is satisfied. The advantage of our optimization strategy is that the continuous solution $\tilde{B}$ we obtained does not depend on the attack budget K. This implies that we can reuse the same $\tilde{B}$ for various K, eliminating the need for recalculations.

However, optimizing the relaxed optimization problem is still challenging because the anomaly score $\mathcal{A}(v)$ in the loss function $\mathcal{L}_a(\tilde{B})$ depends on the variable $\tilde{B}$ in a complex way. After updating $\tilde{B}$, obtaining $\mathcal{A}(v)$ requires iterating over Eqn. 1 dozens of times to get the converged node similarity vector $\vec{s}$, and the gradient needs to be traced back to each iteration. To address this issue, we only iterate over Eqn. 1 once instead of multiple times. The detailed procedures are summarized in Alg. 1 and Fig. 4 (top). Firstly, we update the adjacency matrix with $\tilde{W} = |W - \tilde{B}|$ (line:5), and then we update the similarity score $\mathcal{A}(v)$ based on $\tilde{W}$ for one step using Eqn. 1 and then obtain the anomaly score with Eqn. 3 or 4 (line:6-9). Next, we update attack loss $\mathcal{L}_a(\tilde{B})$ based on $\mathcal{A}(v)$ (line:10), and calculate the projected gradient to optimize $\tilde{B}$ for one step (line: 11-15). Repeating the alternative iteration leads to the convergence of the inner model $\vec{s}$ and also the continuous attack variable $\tilde{B}$. After the iterations, we keep the top-K elements in $\tilde{B}$ to obtain $\bar{B}$ and the others are set to zeros (line:17-18). Finally, the attacked graph is obtained by $\hat{W} = |W - \bar{B}|$ (line:19). This algorithm is also suitable for weighted graphs in which the weights on edges are in [0, 1], and the final solution is to modify K edge weights while the other weights remain unchanged.

2) cf-Attack: While the **alterI**-attack approach is feasible, the one-step update of the inner model is a simple estimation that may not provide accurate attack loss during the iteration. To address this issue and obtain accurate attack loss, we employ the closed-form solution of the inner model to transform the bi-level optimization problem into a single-level problem. According to [48] and [49], the inner model (Eqn. 1) has closed-form solution as follows:

$$\vec{s} = \alpha(I - (1-\alpha)P)^{-1}\vec{r}, \tag{7}$$

where I is an identity matrix. With the closed-form solution, we can directly obtain the accurate anomaly scores after the update of $\tilde{B}$. In contrast to the **alterI**-attack, which iterates the inner model once after updating $\tilde{B}$, our innovative **cf**-attack approach substitutes the Eqn. 1 (line:7) with Eqn. 7 to obtain the accurate connectivity scores $\vec{s}$ for current $\tilde{B}$, and others remain the same.

Algorithm 1 Graph-Space Attack

1: **Input:** Graph with adjacency matrix W, attack budget K, attack iteration T, learning rate η.
2: **Output:** Attacked graph with adjacency matrix $\hat{W}$.
3: **function** ALTERI-ATTACK(W, K, T, η)
4: **for** $t = 1$ to T **do**
5: Update adjacency matrix: $\tilde{W} = |W - \tilde{B}|$.
6: **for** each node v in target set $\mathcal{T}$ **do**
7: Update similarity scores $\vec{s}$ with Eqn. 1.
8: Update anomaly score $\mathcal{A}(v)$ based on $\vec{s}$.
9: **end for**
10: Update objective function $\mathcal{L}_a(\tilde{B})$ with $\mathcal{A}(v)$.
11: **for** each edge $\tilde{b}_{uv}$ in $\tilde{B}$ **do**
12: Calculate gradient $g_{uv} = \tilde{b}_{uv} - \eta \frac{\partial \mathcal{L}(\tilde{B})}{\partial \tilde{b}_{uv}}$
13: Project g_{uv} into [0, 1]
14: Update $\tilde{b}_{uv}$ in $\tilde{B}$
15: **end for**
16: **end for**
17: Choose top-K edges in $\tilde{B}$ to obtain $\bar{B}$:
18:

$$\bar{b}_{uv} = \begin{cases} \tilde{b}_{uv} & \text{if } \tilde{b}_{uv} \in top_K(\tilde{B}), \\ 0 & \text{otherwise.} \end{cases}$$

19: Obtain attacked graph $\hat{W} = |W - \bar{B}|$.
20: **return** $\hat{W}$.
21: **end function**

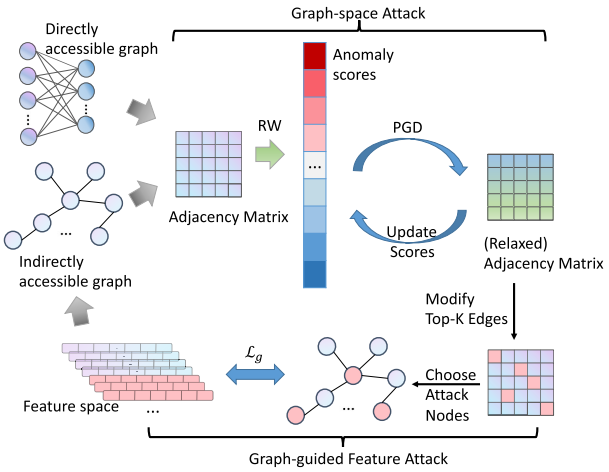


Fig. 4. Illustration of proposed attacks.

C. Complexity Analysis

While **cf**-attack offers a more accurate formulation than **alterI**-attack, it comes with the cost of potential time consumption when calculating the matrix inverse, particularly for graphs with a large number of nodes or edges. In contrast, **alterI**-attack does not encounter such a problem, making it a more efficient option for such scenarios. Both **cf**-attack and **alterI**-attack have their own unique advantages.

Our **alterI**-attack has the complexity of $O(T(|E|+2|V|^2))$. First, we need to update the anomaly scores by Eqn. 1, in which the time complexity is $O(|E|)$ because it transits through all edges in the graph. Our loss function includes a l_2-

norm on the variable $\tilde{B}$, which requires $O(|V|^2)$ computation. Then, we take the gradient for each element in $\tilde{B}$, whose complexity is $O(|V|^2)$. We repeat the process for T steps, then the total complexity for **alterI**-attack is $O(T(|E|+2|V|^2))$. For **cf**-attack, we update the anomaly score by the Eqn. 7, which takes the complexity of $O(|V|^{2.21})$ [50] for sparse matrix inverse. Hence, the total complexity is $O(T(|V|^{2.21}+|E|))$.

VII. Graph-Guided Feature-Space Attacks

A. Motivation for Feature-Space Attacks

Previously, we presented effective graph-space attacks against Di-RWAD. However, for InDi-RWAD, where the graphs are not directly accessible, the *realizability* of the attacks becomes a serious concern: the attacker cannot directly modify the edges in a virtual graph space. Instead, in many practical application scenarios, what the attacker can modify are the attributes associated with the entities in their control. For example, when it comes to network intrusion detection, each TCP connection represents an entity or node, and attackers can manipulate certain TCP connections by altering attributes such as connection duration, protocol type, and the number of urgent packets. Such manipulations will change the structure of the proximity graph in the *ProxGraphRW* model to become perturbed, which can help shield the targeted anomaly TCP connection from being detected.

Thus, investigating feature-space attacks against InDi-RWAD is of significant practical importance. In particular, we consider the scenario where an attacker can manipulate a set of entities (corresponding to nodes in the constructed proximity graph) and modify their features to assist a group of target nodes in avoiding detection. We explore the connection between graph-space and feature-space attacks and demonstrate how guidance from graph-space attacks can be leveraged to construct effective feature-space attacks.

B. Attack Formulation

Consider a set of entities with features $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n]$, where $\mathbf{x}_i \in \mathcal{X}$ denotes the feature vector associated with entity i. As introduced in Section III-B.2, a proximity graph can be constructed from $\mathbf{X}$, where the nodes represent those entities and edges indicate similar node pairs. An attacker aims to allow a set of target entities (nodes) $\mathcal{T}$ to evade detection. We assume that the attacker has control of a set of *attack nodes* $\mathcal{Z}$ such that the features of the nodes in $\mathcal{Z}$ can be arbitrarily modified in a certain domain $\mathcal{X}$. To limit the attacker's ability, we make the restriction that $\mathcal{Z} \cap \mathcal{T} = \emptyset$ and $|\mathcal{Z}| \leq K'$. For an attack node $i \in \mathcal{Z}$, we denote the modified feature vector as $\hat{\mathbf{x}}_i$. The manipulated feature matrix is $\hat{\mathbf{X}}$. We note that since the manipulation of the features leads to the change of graph structure, the anomaly score function $\mathcal{A}(v; \hat{\mathbf{X}})$ depends on the features $\hat{\mathbf{X}}$. Then, we can formulate the feature-space attack as follows:

$$\begin{aligned} \min_{\hat{\mathbf{x}}_i, i \notin \mathcal{T}} \quad & \mathcal{L}(\hat{\mathbf{X}}) = \sum_{v \in \mathcal{T}} \mathbb{I}(\mathcal{A}(v; \hat{\mathbf{X}}) > \theta), \\ \text{s.t.} \quad & \hat{\mathbf{x}}_v = \mathbf{x}_v, \forall v \in \mathcal{T}, \ \hat{\mathbf{x}}_i \in \mathcal{X}, \\ & \mathcal{Z} = \{i | \hat{\mathbf{x}}_i \neq \mathbf{x}_i\}, \ |\mathcal{Z}| \leq K'. \end{aligned} \tag{8}$$

C. Two Levels of Guidance From Graph-Space Attacks

Applying the gradient-descent method to solve problem (8) faces a crucial challenge: while gradient descent can be used to optimize the node features, it is hard to decide which nodes are to be manipulated. In other words, it is nontrivial to guarantee the constraint $|\mathcal{Z}| \leq K'$ while preserving optimization performance. We adopt a divide-and-conquer strategy to tackle this problem: we first select up to K' nodes as the attack nodes and then utilize gradient descent to optimize node features. In particular, we show that the results from graph-space attacks can be innovatively utilized to guide both the selection of attack nodes and feature optimization.

Specifically, given a proximity graph $\mathcal{G}$, we can leverage the attacks in Section VI to produce a poisoned graph $\mathcal{G}'$. Even though $\mathcal{G}'$ might not be directly realized, it represents an excellent candidate in the graph space with which the target nodes $\mathcal{T}$ could evade detection with high probability. Thus, our intuition is to manipulate features so that the resulting proximity graph would approximate $\mathcal{G}'$. To this end, we utilize the guidance from the following two aspects.

a) Guidance on attack node selection: In the graph-space attack, the nodes involved in the structure modification might have a more significant impact on the attack goal. We denote the set of edges/non-edges modified by the attacker as $\mathcal{E}_a$. Intuitively, the modification of $\mathcal{E}_a$ will influence the anomaly scores of the targets most. To preserve such an influence, we set the attack nodes $\mathcal{Z}$ as those ones incident to the edges/non-edges in $\mathcal{E}_a$. Note that we can always easily adjust the budget in the graph-space attack such that the constraint $|\mathcal{Z}| \leq K'$ is satisfied.

After fixing the attack nodes $\mathcal{Z}$, we can follow a similar approach in the graph-space attack to optimize the features. Specifically, we replace the indicator function in (8) with the sum of anomaly scores of target nodes. For discrete features, we relaxed their discrete feature domain to the continuous space denoted by $\tilde{\mathcal{X}}$. Then, let $\tilde{\mathbf{x}}_i \in \tilde{\mathcal{X}}$ denote the relaxed feature, and $\tilde{\mathbf{X}} = \{\tilde{\mathbf{x}}_i | i \in V\}$, the feature-space attack can be formulated as the following optimization problem:

$$\min_{\hat{\mathbf{x}}_i \in \tilde{\mathcal{X}}, i \in \mathcal{Z}} \mathcal{L}_a(\tilde{\mathbf{X}}) = \textstyle\sum_{v \in \mathcal{T}} \mathcal{A}(v; \tilde{\mathbf{X}}). \tag{9}$$

We term this type (with objective function $\mathcal{L}_a$) of feature-space attacks as **G-Guided**. We can straightforwardly adopt the two algorithms **alterI-attack** and **cf-attack** to solve the optimization problem (9), resulting in two variants named **G-Guided-alterI** and **G-Guided-cf**.

b) Guidance on reformulation of attack objective: Beyond the selection of attack nodes, the poisoned graph $\mathcal{G}'$ obtained from the graph-space attack can provide vital information for optimizing the features. Specifically, we aim to optimize the features such that the proximity graph constructed from the modified features would approximate $\mathcal{G}'$ as much as possible. To this end, we reformulate the attack objective function as follows:

$$\mathcal{L}_g(\tilde{\mathbf{X}}) = \sum_{\substack{\{(i,j) | \tilde{b}_{ij} > 0\} \\ i/j \in \mathcal{Z}}} |sim(\mathbf{x}_i, \mathbf{x}_j) - \hat{w}_{i,j}|, \tag{10}$$

where $\hat{w}_{ij}$ is the element in the attacked adjacency matrix $\hat{W}$. This objective function aims to push the similarity between control nodes $\mathbf{x}_i$ and other nodes $\mathbf{x}_j$ (denoted by $sim(\mathbf{x}_i, \mathbf{x}_j)$) close to the manipulated edges $\hat{w}_{i,j}$ in the poisoned graph $\mathcal{G}'$. Intuitively, minimizing $\mathcal{L}_g$ allows us to approximate an inverse problem: given $\mathcal{G}'$, find the node features from which $\mathcal{G}'$ can be constructed. Since (10) is a single-level function, we can directly adopt PGD (similar to the graph-space attack) to solve the optimization problem. We term this type (with objective function $\mathcal{L}_g$) of feature-space attacks as **G-Guided-plus**. The attack algorithm in the feature space is summarized in Alg. 2 and Fig. 4 (bottom).

Algorithm 2 Feature-Space Attack

Input: Feature matrix $\tilde{\mathbf{X}}$, attack nodes $\mathcal{Z}$, attack iteration T, learning rate η.
Output: Attacked feature matrix $\hat{\mathbf{X}}$.
function FEATUREATTACK($\tilde{\mathbf{X}}$, $\mathcal{Z}$, T, η)
  for $t = 1$ to T **do**
    Construct graph based on $\tilde{\mathbf{X}}$ (Section III-B.2).
    Update similarity scores $\vec{s}$ with Eqn. 1.
    Update the anomaly scores based on $\vec{s}$ (Eqn. 4).
    Update objective function $\mathcal{L}(\tilde{\mathbf{X}})$.
    for each attack nodes $\tilde{\mathbf{x}}_i, i \in \mathcal{Z}$ **do**
      Calculate gradient $g_i = \tilde{\mathbf{x}}_i - \eta \frac{\partial \mathcal{L}(\tilde{\mathbf{X}})}{\partial \tilde{\mathbf{x}}_i}$.
      Project $g_{i,j}$ into the feasible set $\tilde{\mathcal{X}}$.
      Update $\tilde{\mathbf{x}}_{i,j}$ in $\tilde{\mathbf{X}}$.
    end for
  end for
  Rounding the attacked feature:

$$\hat{\mathbf{x}}_i = \begin{cases} round(\tilde{\mathbf{x}}_{ij}) & \text{if feature } j \text{ is discrete,} \\ \tilde{\mathbf{x}}_{ij} & \text{otherwise.} \end{cases}$$

  return Attacked feature matrix $\hat{\mathbf{X}}$.
end function

The perturbed graph obtained from the graph-space attack serves as a valuable source of information. It not only highlights the crucial nodes that should be targeted by the feature-space attack but also suggests how the node features should be modified to maximize the impact on the anomaly score calculation.

VIII. EXPERIMENTS

In this section, we evaluate the performances of our proposed attacks by answering these four major questions: 1) Are our proposed graph-space attacks effective? 2) What are the preferences of the proposed graph attack? 3) How effective are the graph-guided feature-space attacks? 4) How is the transferability of the graph-guided feature-space attacks?

A. Datasets and Experiment Settings

We consider four datasets that are commonly used for graph-based anomaly detection: Paper-Author, Magazine, KDD-99, and MINIST (outlier). Among them, the first two are bipartite graphs while the latter two datasets are feature data. Below is the detailed description. **All datasets, source code for our proposed attacks, and evaluated baselines are in our *GitHub* link.**[1]

- Paper-Author [10]: This dataset contains papers crawled from the arXiv preprint database. Nodes U represent papers, while nodes V represent authors. An edge $\langle u, v \rangle$ indicates that the author v is shown in the paper u. We randomly sample $10,000$ records and delete nodes with degrees lower than 5, resulting in $|U|, |V| = 2311, 405$. We manually inject 10% of anomaly nodes following [34].
- Magazine: This dataset contains Amazon Reviews Data[2] under the category of Magazine Subscriptions. We randomly sample $100,000$ records and removed nodes with degrees lower than 3, resulting in $|U|, |V| = 1079, 1180$ nodes. We also inject 10% of anomaly nodes manually following [34].
- KDD-99 [15]: The dataset contains network intrusion data with 41 features and 4 types of attacks. We randomly sample $10,000$ benign data and 100 anomaly data for the experiment.
- MINIST (outlier): This is a subset of the MINST handwritten digits dataset, created for the outlier detection task in Outlier Detection DataSets.[3] It contains a total of 7603 images, with 6903 images of digit-0 regarded as normal points and 700 images of digit-6 regarded as outliers. Each sample has 100 features.

B. Experimental Settings

We conduct our experiments on Ubuntu 20.04 system with an NVIDIA GeForce RTX 3090 GPU, Python 3.7, and PyTorch 1.10.0. All the experiments are repeated 10 times with different random seeds, and different target nodes are sampled.

1) Target Nodes and Budgets: For attacking *BiGraphRW* model, we sample 5 target nodes from the top 100 anomaly nodes, while in *ProxGraphRW* model, we sample 20 target nodes from the top 100 anomaly nodes. We set the attack edge budget proportion to the sum of *target node degrees* (e.g., budget 10% : $K = 0.1 \times \sum_{v \in \mathcal{T}} d(v)$, where $d(v)$ is the degree of node v). Setting the budget associated with node degree is commonly adopted in targeted attacks such as Nettack [22], [42]. In feature-space attacks, we set the number of attack nodes $\mathcal{Z}$ as the number of nodes involved in the **alterI** graph-space attack. Then the attack intensity is associated with the attack budget in the graph space attack.

2) Evaluation Metrics: Our main focus is to evaluate the effectiveness of our proposed method facilitating target nodes to evade detection under different detection thresholds. Usually, the detection threshold θ is set to the proportion of data size, and we evaluate the level of detect ratio as the top 5% and 10%. We then use the *evasion rate* ER of target nodes under these detection thresholds as the main metric. Specifically, the evasion rate is computed as $\mathsf{ER} = n_0/|\mathcal{T}|$, where n_0 is the number of target nodes not shown in the top 5% or 10%

[1] https://github.com/Yuni-Lai/CoupledAttackRW
[2] https://nijianmo.github.io/amazon/, accessed May 2023
[3] http://odds.cs.stonybrook.edu/, accessed May 2023

anomaly scores (i.e., evaded successfully). Besides, we also evaluate the average anomaly scores of target nodes.

3) Baselines: We evaluate the effectiveness of our proposed attacks against several baselines for both graph-space attacks and feature-space attacks.

a) Graph-space attack: The most relevant prior work is [42]. Although this work also proposes a targeted attack for the RW model, it is specific to the DeepWalk model and cannot be directly applied to our RWAD systems. Therefore, we transfer its targeted attack to our model. Besides, we also adopt two common baselines RndAdd and DegAdd following [42].

- **RndAdd**: This baseline randomly adds candidate edges, where the candidate edges are the edges incident to target nodes.
- **DegAdd**: This baseline adds candidate edges with the top-K highest degrees, where the candidate edges are also the edges incident to target nodes.
- **DeepWalk** [42]: In this baseline, we transfer the attack designed for DeepWalk to RWAD models.
- Our methods: **alterI** and **cf** are our proposed attacks with alternative iteration and closed-form solution, respectively.

b) Feature-space attack: To evaluate the effectiveness of our graph-guided attack in node selection, we include random selection as a baseline for comparison.

- **VanillaOpt**: This baseline randomly selects attack nodes from candidates and optimizes node features with the objective function $\mathcal{L}_a(\tilde{\mathbf{X}})$ in (9) with strategy **alterI**.
- Our methods: We use the graph-space attacks to guide the selection of attack nodes and choose $\mathcal{L}_a(\tilde{\mathbf{X}})$ as the attack objective function, resulting in two attack methods **G-guided-alterI** and **G-guided-cf**, which adopt **alterI** and **cf** to optimize node features respectively. In addition, when the objective function $\mathcal{L}_g(\tilde{\mathbf{X}})$ (10) is selected, the attack method is **G-guided-plus**.

4) Hyper-Parameters: Grid search is employed to find the optimal hyper-parameters in all the attack methods over different datasets. For *BiGraphRW* model, the regularization parameter $\lambda = 1 \times 10^{-6}$, learning rate $lr = 1.0$, 60 epochs with SGD optimizer. For *ProxGraphRW* model, we evaluate proximity graphs constructed with cosine similarity ($Cos(\mathbf{x}_i, \mathbf{x}_j) = \frac{<\mathbf{x}_i, \mathbf{x}_j>}{||\mathbf{x}_i|| ||\mathbf{x}_j||}$) and correlation similarity ($Corr(\mathbf{x}_i, \mathbf{x}_j) = \frac{<\mathbf{x}_i - \bar{\mathbf{x}}_i, \mathbf{x}_j - \bar{\mathbf{x}}_j>}{||\mathbf{x}_i - \bar{\mathbf{x}}_i|| ||\mathbf{x}_j - \bar{\mathbf{x}}_j||}$). The similarity threshold ϵ for constructing the graph is 0.8 for the KDD-99 dataset and 0.5 for the MNIST dataset; We employed the regularization parameter $\lambda = 1 \times 10^{-4}$, learning rate $lr = 1.0$, 35 epochs for the KDD-99 dataset and 100 for the MNIST dataset with Adam optimizer in the graph-space attack. In feature-space attack, learning rate $lr = 1.0$, 500 epochs with Adam optimizer.

C. Performance of Graph-Space Attacks

To begin with, we evaluate the performance of the target RWAD models over corresponding datasets. As shown in Tab. II, both models achieved an AUC (area under reception curve) of at least 0.89, demonstrating a strong ability to identify anomalies.

TABLE II
AUC OF RWAD

Models	*BiGraphRW*		*ProxGraphRW*	
Dataset	Author-Paper	Magazine	KDD-99	MNIST
AUC	1.00	0.89	0.98	0.90

TABLE III
GRAPH ATTACK RESULTS ON *BiGraphRW* MODEL

Dataset	Metrics	budget	RndAdd	DegAdd	DeepWalk	alterI	cf
Author-Paper	ER (5%)	0%	0.560	0.560	0.560	0.560	0.560
		20%	0.560	0.560	0.578	0.720	**0.760**
		40%	0.560	0.560	0.578	0.880	**0.940**
		60%	0.580	0.560	0.578	0.920	**0.960**
		80%	0.580	0.560	0.600	0.980	**1.000**
		100%	0.580	0.560	0.600	**1.000**	**1.000**
	ER (10%)	0%	0.000	0.000	0.000	0.000	0.000
		20%	0.000	0.000	0.000	0.060	**0.280**
		40%	0.000	0.000	0.000	0.260	**0.360**
		60%	0.000	0.000	0.000	**0.460**	0.360
		80%	0.000	0.000	0.000	**0.660**	0.600
		100%	0.000	0.000	0.000	**0.820**	0.740
Magzine	ER (5%)	0%	0.740	0.740	0.740	0.740	0.740
		20%	0.760	0.740	0.760	0.760	**0.780**
		40%	0.760	0.760	0.760	**0.880**	0.860
		60%	0.760	0.760	0.760	**0.920**	0.880
		80%	0.760	0.760	0.760	**0.960**	0.880
		100%	0.780	0.760	0.760	**0.980**	0.880
	ER (10%)	0%	0.380	0.380	0.380	0.380	0.380
		20%	0.380	0.380	0.380	0.500	**0.600**
		40%	0.400	0.380	0.380	0.560	**0.740**
		60%	0.400	0.380	0.400	0.620	**0.760**
		80%	0.400	0.380	0.400	0.760	**0.820**
		100%	0.400	0.400	0.400	0.840	**0.860**

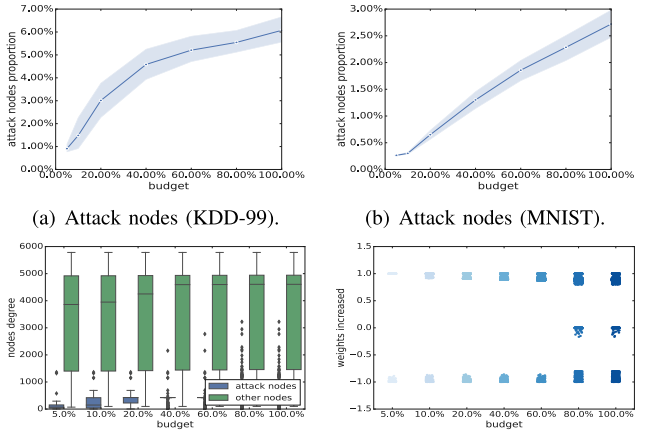
1) Effectiveness of Attacks: We present the evasion rates ER of those attack methods under different detection levels (top-5%/10%) in Tab. III and IV. We observe that our proposed graph attack methods, **alterI** and **cf**, significantly outperform other baselines on all datasets. For instance, at the detection level of top-5%, our results indicate that our proposed attack on *BiGraphRW* model is highly effective, achieving an evasion rate of over 85% with a budget of 40.0%. Similarly, for *ProxGraphRW* model, with a budget of 60.0%, the evasion rate (under detection threshold top-10%) is over 80% on the MNIST dataset. Since the MNIST dataset is relatively easier to attack, we report the attack performance at a higher detection threshold. The reason why the **DeepWalk** method does not exhibit a strong attack effect could be attributed to its transferability across different types of random walk models.

Comparing **alterI** and **cf** attack, it was observed that **cf** attack slightly outperforms **alterI** in most cases. In our experiments, we observe that **cf** can achieve significantly lower attack loss in the continuous domain (i.e., $\tilde{B}$). However, when discretizing the optimization results, the attack performance is not guaranteed to be preserved. While **cf** is generally more effective (also observed for feature-space attacks in Section VIII-D), **alterI** is more efficient on larger graphs such as KDD-99 and MNIST (see Tab. V).

2) Preferences of Graph Attack: We further present a more detailed analysis of the graph attack results in Fig. 5, in which Fig. 5(a) and 5(b) show the proportion of the attacked nodes

TABLE IV
GRAPH ATTACK RESULTS ON *ProxGraphRW* MODEL

Dataset	Similarity	budget	RndAdd	DegAdd	DeepWalk	alterI	cf
KDD99 ER (5%)	cosine	0%	0.045	0.045	0.045	0.045	0.045
		10%	0.045	0.045	0.045	0.050	**0.055**
		20%	0.045	0.045	0.045	0.155	**0.245**
		40%	0.045	0.045	0.045	0.605	**0.620**
		60%	0.045	0.045	0.045	0.745	**0.825**
		80%	0.055	0.045	0.050	0.775	**0.865**
		100%	0.085	0.045	0.060	0.775	**0.875**
	correlation	0%	0.045	0.045	0.045	0.045	0.045
		10%	0.045	0.045	0.045	0.055	**0.060**
		20%	0.045	0.045	0.045	0.110	**0.150**
		40%	0.045	0.045	0.045	0.315	**0.405**
		60%	0.045	0.045	0.045	0.575	**0.690**
		80%	0.050	0.045	0.050	0.670	**0.735**
		100%	0.060	0.045	0.055	0.695	**0.845**
MNIST ER (10%)	cosine	0%	0.000	0.000	0.000	0.000	0.000
		10%	0.000	0.000	0.000	**0.060**	0.045
		20%	0.000	0.000	0.000	**0.210**	0.135
		40%	0.000	0.000	0.000	**0.585**	0.515
		60%	0.000	0.000	0.020	0.800	**0.860**
		80%	0.005	0.000	0.030	0.940	**0.975**
		100%	0.050	0.000	0.070	0.985	**0.995**
	correlation	0%	0.000	0.000	0.000	0.000	0.000
		10%	0.000	0.000	0.000	0.045	**0.060**
		20%	0.000	0.000	0.000	**0.205**	0.185
		40%	0.000	0.000	0.000	**0.555**	0.550
		60%	0.010	0.000	0.010	0.770	**0.825**
		80%	0.040	0.000	0.045	**0.940**	**0.940**
		100%	0.095	0.005	0.080	**0.995**	**0.995**

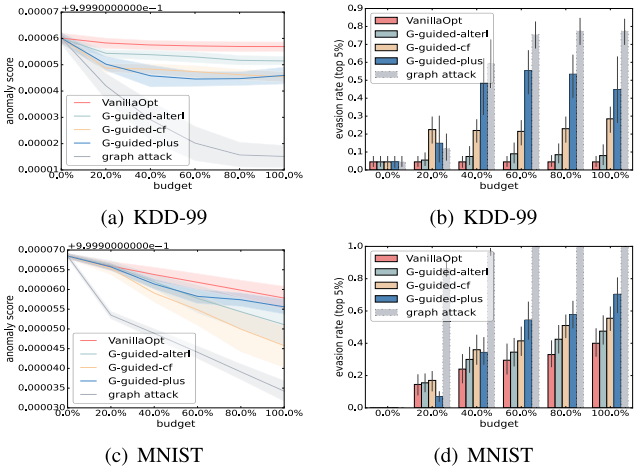


(a) Attack nodes (KDD-99). (b) Attack nodes (MNIST).

(c) The attacker prefers nodes with lower degrees. (d) The weights changed by attacker.

(e) The attacker prefers adding edges between target nodes and the others. (f) The attacker increases the degree of target nodes.

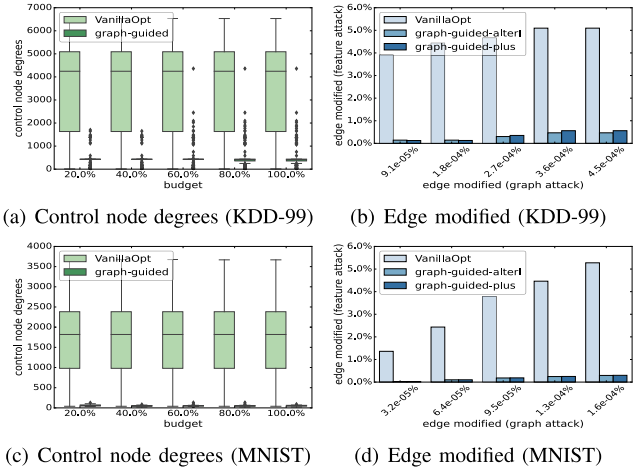
Fig. 5. Graph-space attack (**alterI**) result analysis on KDD-99.



(a) KDD-99 (b) KDD-99

(c) MNIST (d) MNIST

Fig. 6. Feature-space attack results.



(a) Control node degrees (KDD-99) (b) Edge modified (KDD-99)

(c) Control node degrees (MNIST) (d) Edge modified (MNIST)

Fig. 7. Result analysis of feature-space attacks.

(to the total number of nodes) corresponding to different budgets. On average, only about $1\% - 6\%$ (KDD-99) and $0.3\% - 2.7\%$ (MNIST) of nodes are involved in the edge modification under various budgets (Fig. 5(b)). In Fig. 5(c), we present the node degrees of attack nodes and others, and we observe that the attacker prefers nodes with lower degrees as attack nodes. Fig. 5(d) presents the weights changed in the attack. We observe that the attacker tends to make larger weight changes in the K attack edges. This is because we choose the top-K edges in priority of the values in $\tilde{B}$, and a higher value of $\tilde{b}_{ij}$ leads to larger weight change. The attacker mainly adds/deletes edges between target nodes and other nodes (Fig. 5(e)), and the target-other edge modification tends to increase the degree of target nodes (Fig. 5(f)). These actually provide convenience for our graph-guided feature attack with attack loss $\mathcal{L}_g(\tilde{\mathbf{X}})$, where the target node features are fixed (the edges between target-target are fixed) and the attack nodes can be optimized to be close to the desired edge weights (the edges between target nodes and control nodes). We observe similar phenomena in the MNIST dataset.

D. Performances of Graph-Guided Feature-Space Attacks

First of all, to limit the attack intensity, we set the number of control nodes $|\mathcal{Z}|$ as the number of nodes involved in the graph-space attack. As mentioned before in Section VIII-C.2, the number of nodes involved in the graph-space attack ranges from $1\% - 6\%$ in the KDD-99 dataset and $0.3\% - 2.7\%$ in the MNIST dataset with various edge manipulation budgets.

TABLE V
RUNTIME COMPARISON OF **ALTERI** AND **CF**-ATTACK

	Attacks	Author-Paper	Magazine	KDD-99	MNIST
Graph attack	alterI	00:00:07	00:00:04	00:00:10	00:00:16
	cf	00:00:02	00:00:02	00:00:23	00:00:35
Feature attack	alterI	-	-	00:00:27	00:00:18
	cf	-	-	00:03:26	00:01:49

1) Effectiveness of Attacks: We compare the performances of these feature-space attacks in Fig. 6. Our analysis shows that **G-guided-alterI** outperforms the **VanillaOpt** method, achieving much lower anomaly scores and higher evasion rates. These two models are only different in the selection of attack nodes, which indicates the effectiveness of using guidance from graph-space attacks in node selection. Comparing the performance of the **alterI** and **cf** attack strategies under $\mathcal{L}_a$, we observe that **cf**-attack also improves the performance, although the side effect is that **cf**-attack takes about 7 times longer than **alterI** in our experiments (Tab. V). Additionally, **G-guided-plus** has a higher evasion rate than **G-guided-alterI** and **G-guided-cf** in most cases, indicating the advantage of using the attack loss $\mathcal{L}_g$ as further guidance for feature attack.

2) Unnoticeability of Attack: In Fig. 7, we provide an analysis of the feature attack highlighting its advantage of unnoticeability. Specifically, we visualize the control nodes' original degree in Fig. 7(a) and Fig. 7(c). The manipulation in the feature space will then lead to the perturbation in the graph space. In Fig. 7(b) and Fig. 7(d), in order to quantify the perturbation volume, we present the ratio of edges modified by the feature-space attack on the y-axis and the ratio of edges modified by the graph-space attack x-axis. These ratios are proportionate to the original graph's total number of edges. As mentioned earlier, the graph attack prefers the attack nodes with lower degrees. As a result, our graph-guided attack nodes have lower node degrees compared to **VanillaOpt** (Fig. 7(a) and Fig. 7(c)). This leads to significantly fewer edge modifications in graph-guided attacks compared to **VanillaOpt** (Fig. 7(b) and Fig. 7(d)), which enhances the unnoticeability of the attack. In particular, both of our graph-guided attacks only lead to less than 0.5% edge modification in the graph space in both datasets.

E. Transferability of Graph-Guided Attack

We transfer our feature-space attacks to several unsupervised anomaly detection models, including Beta-VAE [51], IForest [52], and ECOD [53]. Tab. VI shows the anomaly scores of target nodes before and after the transfer attack based on our **G-guided-alterI** and **G-guided-plus** feature attack on the KDD-99 dataset. The results indicate that the graph-guided attack with graph attack loss significantly decreases the anomaly scores of the target nodes across different models. This suggests that the graph-guided attack on RWAD has the potential to be used as a surrogate model for black-box attacks. The graph-guided attack could be a useful tool for attackers to evade detection and deceive anomaly detection systems in real-world scenarios.

TABLE VI
TRANSFERABILITY: THE CHANGE IN ANOMALY SCORE (%) COMPARED TO THE CLEAN DATA. LOWER IS BETTER

Detect Methods	Attack Methods	20%	40%	60%	80%	100%
Beta-VAE	VanillaOpt	-11.56	-13.97	-14.70	-15.18	-15.68
	G-guided-alterI	-4.15	-5.65	-6.13	-7.14	-9.711
	G-guided-plus	**-25.26**	**-31.79**	**-33.25**	**-33.94**	**-33.99**
IForest	VanillaOpt	-10.63	-0.06	-8.41	-0.82	-11.93
	G-guided-alterI	9.94	10.44	-0.18	0.39	-2.31
	G-guided-plus	**-25.03**	**-44.21**	**-40.27**	**-47.58**	**-47.26**
ECOD	VanillaOpt	-2.20	-2.72	-2.90	-2.988	-3.099
	G-guided-alterI	-0.29	-0.64	-0.66	-0.90	-1.28
	G-guided-plus	**-3.70**	**-5.32**	**-5.81**	**-6.01**	**-6.00**

TABLE VII
PROXGRAPHRW MODEL DETECTION PERFORMANCE WITH VARIOUS FEATURE SIMILARITY THRESHOLD ϵ (CORRELATION SIMILARITY)

KDD-99		MNIST	
ϵ	AUC	ϵ	AUC
0.5	0.97	**0.5**	**0.90**
0.6	0.97	0.6	0.79
0.7	0.96	0.7	0.60
0.8	**0.98**	0.8	0.44
0.9	0.87	0.9	0.45

IX. LIMITATION AND FUTURE WORK

Although our paper introduces coupled-space attacks for RWAD and demonstrates the superior performance of our proposed methods compared to the baselines, we did identify certain limitations. Specifically, in Fig. 6, we observed that our feature-space attacks were not as effective as our graph-space attacks. Additionally, the unnoticeability of feature attacks was found to be comparatively weaker than graph attacks. Fig. 7 indicates that the feature space attack resulted in a higher proportion of graph structure perturbation.

These limitations highlight areas for further investigation and improvement in future research. While our proposed coupled-space attacks offer significant advancements, addressing these limitations could potentially enhance the effectiveness and stealthiness of feature-space attacks in RWAD.

Generalization of coupled-space attack: In this study, we introduce coupled-space attacks against RWAD, where the interdependency between the graph space and the feature space is exploited to enhance the effectiveness of attacks. Besides RWAD, there are many other feature-derived graph models where the graph and feature are interdependent [54], [55]. For example, graph structure can be constructed based on traffic sensor data [56], earthquake sensor data [56], image data [57], video data [58], and genomics data [59]. Future work can generalize our proposed strategies to more feature-derived graph-based models in which the graph is constructed on raw features. Because the model directly relies on the graph, as long as the graph constructed on the perturbed feature is close to the perturbed graph, the attack is expected to be effective.

In this paper, we show the potential of transferring our graph-guided feature-space attacks on RWAD to three unsupervised anomaly detection models. Future research can extend this work to apply RWAD for black-box attacks on other deep

learning-based anomaly detection systems, without relying on labeled data or inner models.

X. Conclusion

In conclusion, this paper has shed light on the vulnerabilities of Random-Walk-based Anomaly Detection (RWAD), a classical and important anomaly detection tool. Specifically, we introduce a novel study of adversarial poisoning attacks on RWAD, where the graph is constructed on top of the feature space. We provide a theoretical understanding of these attacks, including proof of NP-hardness. Our approach involves proposing graph-space attacks and using the graph attack to guide the feature-space attack, which bridges the gap between these two attacks. Our experiments on four datasets, encompassing both directly and indirectly accessible graphs, demonstrate the effectiveness of our proposed graph-space attack and its ability to guide the selection of attack nodes and optimization of the attack loss for feature-space attacks. By taking RWAD as an example, our study provides valuable insights into the effectiveness of graph-space attacks and feature-space attacks.

Appendix

We present more experimental results in this appendix. We evaluate the impact of the feature similarity threshold ϵ (hyper-parameter) ranging from $0.5 - 0.9$ in Table VII. In our experiment, we set the ϵ with the best detection performance.

References

[1] J. Tang, F. Hua, Z. Gao, P. Zhao, and J. Li, "GADBench: Revisiting and benchmarking supervised graph anomaly detection," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 36, 2024, pp. 1–26.

[2] L. Page, S. Brin, R. Motwani, and T. Winograd, "The pagerank citation ranking: Bringing order to the web," Stanford InfoLab, Stanford, CA, USA, Tech. Rep. SIDL-WP-1999-0120, 1999.

[3] C. Oliveira, J. Torres, M. I. Silva, D. Aparício, J. T. Ascensão, and P. Bizarro, "GuiltyWalker: Distance to illicit nodes in the Bitcoin network," 2021, *arXiv:2102.05373*.

[4] X. Li, Y. Zhuang, Y. Fu, and X. He, "A trust-aware random walk model for return propensity estimation and consumer anomaly scoring in online shopping," *Sci. China Inf. Sci.*, vol. 62, p. 52101, Mar. 2019.

[5] H. Yu, M. Kaminsky, P. B. Gibbons, and A. Flaxman, "SybilGuard: Defending against Sybil attacks via social networks," in *Proc. Conf. Appl., Technol., Architectures, Protocols Comput. Commun.*, Aug. 2006, pp. 267–278.

[6] H. Yu, P. B. Gibbons, M. Kaminsky, and F. Xiao, "SybilLimit: A near-optimal social network defense against Sybil attacks," in *Proc. IEEE Symp. Secur. Privacy*, May 2008, pp. 3–17.

[7] L. Shi, S. Yu, W. Lou, and Y. T. Hou, "SybilShield: An agent-aided social network-based Sybil defense among multiple communities," in *Proc. IEEE INFOCOM*, Apr. 2013, pp. 1034–1042.

[8] W. Wei, F. Xu, C. C. Tan, and Q. Li, "SybilDefender: A defense mechanism for Sybil attacks in large social networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 24, no. 12, pp. 2492–2502, Dec. 2013.

[9] J. Jia, B. Wang, and N. Z. Gong, "Random walk based fake account detection in online social networks," in *Proc. 47th Annu. IEEE/IFIP Int. Conf. Dependable Syst. Netw. (DSN)*, Jun. 2017, pp. 273–284.

[10] J. Sun, H. Qu, D. Chakrabarti, and C. Faloutsos, "Neighborhood formation and anomaly detection in bipartite graphs," in *Proc. 5th IEEE Int. Conf. Data Mining (ICDM)*, Nov. 2005, p. 8.

[11] E. Goodman, J. Ingram, S. Martin, and D. Grunwald, "Using bipartite anomaly features for cyber security applications," in *Proc. IEEE 14th Int. Conf. Mach. Learn. Appl. (ICMLA)*, Dec. 2015, pp. 301–306.

[12] H. Cheng, P.-N. Tan, C. Potter, and S. Klooster, "Detection and characterization of anomalies in multivariate time series," in *Proc. SIAM Int. Conf. Data Mining*, Apr. 2009, pp. 413–424.

[13] H. Ren, M. Liu, Z. Li, and W. Pedrycz, "A piecewise aggregate pattern representation approach for anomaly detection in time series," *Knowl.-Based Syst.*, vol. 135, pp. 29–39, Nov. 2017.

[14] Z. Yao, P. Mark, and M. Rabbat, "Anomaly detection using proximity graph and PageRank algorithm," *IEEE Trans. Inf. Forensics Security*, vol. 7, no. 4, pp. 1288–1300, Aug. 2012.

[15] H. D. K. Moonesinghe and P. N. Tan, "Outlier detection using random walks," in *Proc. 18th IEEE Int. Conf. Tools Artif. Intell. (ICTAI)*, Nov. 2006, pp. 532–539.

[16] G. Pang, L. Cao, and L. Chen, "Outlier detection in complex categorical data by modeling the feature value couplings," in *Proc. Int. Joint Conf. Artif. Intell. (IJCAI)*, 2016, pp. 1–8.

[17] C. Wang, H. Gao, Z. Liu, and Y. Fu, "A new outlier detection model using random walk on local information graph," *IEEE Access*, vol. 6, pp. 75531–75544, 2018.

[18] C. Wang, Z. Liu, H. Gao, and Y. Fu, "Applying anomaly pattern score for outlier detection," *IEEE Access*, vol. 7, pp. 16008–16020, 2019.

[19] V. N. Ioannidis, D. Berberidis, and G. B. Giannakis, "Unveiling anomalous nodes via random sampling and consensus on graphs," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Jun. 2021, pp. 5499–5503.

[20] T. Zhao, C. Deng, K. Yu, T. Jiang, D. Wang, and M. Jiang, "Error-bounded graph anomaly loss for GNNs," in *Proc. 29th ACM Int. Conf. Inf. Knowl. Manag.*, 2020, pp. 1873–1882.

[21] J. He, Q. Xu, Y. Jiang, Z. Wang, and Q. Huang, "ADA-GAD: Anomaly-denoised autoencoders for graph anomaly detection," in *Proc. AAAI Conf. Artif. Intell.*, 2024, vol. 38, no. 8, pp. 8481–8489.

[22] D. Zügner, A. Akbarnejad, and S. Günnemann, "Adversarial attacks on neural networks for graph data," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2018, pp. 2847–2856.

[23] D. Zügner and S. Günnemann, "Adversarial attacks on graph neural networks via meta learning," in *Proc. Int. Conf. Learn. Represent.*, 2019, pp. 1–15. [Online]. Available: https://openreview.net/forum?id=Bylnx209YX

[24] Y. Zhu et al., "BinarizedAttack: Structural poisoning attacks to graph-based anomaly detection," in *Proc. IEEE 38th Int. Conf. Data Eng. (ICDE)*, May 2022, pp. 14–26.

[25] K. Zhou, T. P. Michalak, M. Waniek, T. Rahwan, and Y. Vorobeychik, "Attacking similarity-based link prediction in social networks," in *Proc. 18th Int. Conf. Auton. Agents MultiAgent Syst.*, 2019, pp. 305–313.

[26] K. Zhou and Y. Vorobeychik, "Robust collective classification against structural attacks," in *Proc. Conf. Uncertainty Artif. Intell.*, 2020, pp. 250–259.

[27] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," 2014, *arXiv:1412.6572*.

[28] N. Carlini and D. Wagner, "Towards evaluating the robustness of neural networks," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2017, pp. 39–57.

[29] S.-M. Moosavi-Dezfooli, A. Fawzi, O. Fawzi, and P. Frossard, "Universal adversarial perturbations," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 1765–1773.

[30] B. Wang and N. Z. Gong, "Attacking graph-based classification via manipulating the graph structure," in *Proc. 26th ACM SIGSAC Conf. Comput. Commun. Security (CCS)*, 2019, pp. 2023–2040.

[31] C. You, D. P. Robinson, and R. Vidal, "Provable self-representation based outlier detection in a union of subspaces," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 3395–3404.

[32] L. Akoglu, M. McGlohon, and C. Faloutsos, "Oddball: Spotting anomalies in weighted graphs," in *Proc. 14th Pacific-Asia Conf. Adv. Knowl. Discovery Data Mining (PAKDD)*, Hyderabad, India. Berlin, Germany: Springer, 2010, pp. 410–421.

[33] Q. Ding, N. Katenka, P. Barford, E. Kolaczyk, and M. Crovella, "Intrusion as (anti)social communication: Characterization and detection," in *Proc. 18th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2012, pp. 886–894.

[34] B. Hooi, H. A. Song, A. Beutel, N. Shah, K. Shin, and C. Faloutsos, "FRAUDAR: Bounding graph fraud in the face of camouflage," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2016, pp. 895–904.

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

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

[37] S. Bandyopadhyay, S. V. Vivek, and M. Murty, "Outlier resistant unsupervised deep architectures for attributed network embedding," in *Proc. 13th Int. Conf. Web Search Data Mining*, 2020, pp. 25–33.

[38] K. Ding, J. Li, R. Bhanushali, and H. Liu, "Deep anomaly detection on attributed networks," in *Proc. SIAM Int. Conf. Data Mining*, 2019, pp. 594–602.

[39] H. Qiao and G. Pang, "Truncated affinity maximization: One-class homophily modeling for graph anomaly detection," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 36, 2024, pp. 1–23.

[40] Y. Liu, K. Ding, Q. Lu, F. Li, L. Y. Zhang, and S. Pan, "Towards self-interpretable graph-level anomaly detection," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 36, 2024, pp. 1–13.

[41] A. Roy et al., "GAD-NR: Graph anomaly detection via neighborhood reconstruction," in *Proc. 17th ACM Int. Conf. Web Search Data Mining*, 2024, pp. 576–585.

[42] A. Bojchevski and S. Günnemann, "Adversarial attacks on node embeddings via graph poisoning," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 695–704.

[43] H. Chang et al., "Adversarial attack framework on graph embedding models with limited knowledge," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 5, pp. 4499–4513, May 2023.

[44] S. Zhang, H. Chen, X. Sun, Y. Li, and G. Xu, "Unsupervised graph poisoning attack via contrastive loss back-propagation," in *Proc. ACM Web Conf.*, 2022, pp. 1322–1330.

[45] L. Lin, E. Blaser, and H. Wang, "Graph structural attack by perturbing spectral distance," in *Proc. 28th ACM SIGKDD Conf. Knowl. Discovery Data Mining*, Aug. 2022, pp. 989–998.

[46] V. W. Anelli, Y. Deldjoo, T. DiNoia, and F. A. Merra, "Adversarial recommender systems: Attack, defense, and advances," in *Recommender Systems Handbook*. USA: Springer, 2021, pp. 335–379.

[47] N. Perra and S. Fortunato, "Spectral centrality measures in complex networks," *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 78, no. 3, Sep. 2008, Art. no. 036107.

[48] P. Boldi, M. Santini, and S. Vigna, "A deeper investigation of pagerank as a function of the damping factor," in *Proc. Dagstuhl Seminar*. Wadern, Germany: Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2007.

[49] J. Gasteiger, A. Bojchevski, and S. Günnemann, "Combining neural networks with personalized pagerank for classification on graphs," in *Proc. Int. Conf. Learn. Represent.*, 2019.

[50] S. Casacuberta and R. Kyng, "Faster sparse matrix inversion and rank computation in finite fields," in *Proc. 13th Innov. Theor. Comput. Sci. Conf. (ITCS)*. Wadern, Germany: Schloss-Dagstuhl-Leibniz Zentrum für Informatik, 2022, pp. 1–26.

[51] C. P. Burgess et al., "Understanding disentangling in β-VAE," 2018, *arXiv:1804.03599*.

[52] F. T. Liu, K. M. Ting, and Z.-H. Zhou, "Isolation forest," in *Proc. 8th IEEE Int. Conf. Data Min.*, Dec. 2008, pp. 413–422.

[53] Z. Li, Y. Zhao, X. Hu, N. Botta, C. Ionescu, and G. Chen, "ECOD: Unsupervised outlier detection using empirical cumulative distribution functions," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 12, pp. 12181–12193, Dec. 2023.

[54] T. T. Müller et al., "A survey on graph construction for geometric deep learning in medicine: Methods and recommendations," *Trans. Mach. Learn. Res.*, pp. 1–35, Jan. 2024.

[55] L. Qiao, L. Zhang, S. Chen, and D. Shen, "Data-driven graph construction and graph learning: A review," *Neurocomputing*, vol. 312, pp. 336–351, Oct. 2018.

[56] S. Bloemheuvel, J. van den Hoogen, and M. Atzmueller, "Graph construction on complex spatiotemporal data for enhancing graph neural network-based approaches," *Int. J. Data Sci. Anal.*, vol. 18, pp. 157–174, Sep. 2023.

[57] S. Li, W. Yao, Y. Gao, Y. Ma, and B. Liu, "Progressive structure enhancement graph convolutional network for face clustering," *Eng. Appl. Artif. Intell.*, vol. 127, Jan. 2024, Art. no. 107274.

[58] X. Wang and A. Gupta, "Videos as space-time region graphs," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 399–417.

[59] Y. Qian, P. Expert, P. Panzarasa, and M. Barahona, "Geometric graphs from data to aid classification tasks with graph convolutional networks," *Patterns*, vol. 2, no. 4, Apr. 2021, Art. no. 100237.