Anomaly Detection on Static and Dynamic Graphs using Graph Convolutional Neural Networks

Amani Abou Rida[1*], Rabih Amhaz[1,2†] and Pierre Parrend[1,3†]

[1*]ICube (Laboratoire des sciences de l'ingénieur, de l'informatique et de l'imagerie), UMR 7357, Université de Strasbourg, CNRS, 67000 Strasbourg, France.
[2]ECAM Strasbourg-Europe, 67300 Schiltigheim, France.
[3]EPITA, 5, Rue Gustave Adolphe Hirn, 67000 Strasbourg, France.

*Corresponding author(s). E-mail(s): abou-rida.amani@etu.unistra.fr;
Contributing authors: amhaz@unistra.fr; pierre.parrend@epita.fr;
†These authors contributed equally to this work.

Abstract

Anomalies represent rare observations that vary significantly from others. Anomaly detection intended to discover these rare observations has the power to prevent detrimental events, such as financial fraud, network intrusion, and social spam. However, conventional anomaly detection methods cannot handle this problem well because of the complexity of graph data (e.g., irregular structures, relational dependencies, node/edge types/attributes/directions/multiplicities/weights, large scale, etc.) [1]. Thanks to the rise of deep learning in solving these limitations, graph anomaly detection with deep learning has obtained an increasing attention from many scientists recently. However, while deep learning can capture unseen patterns of multi-dimensional Euclidean data, there is a huge number of applications where data are represented in the form of graphs. Graphs have been used to represent the structural relational information, which raises the graph anomaly detection problem - identifying anomalous graph objects (i.e., vertex, edges, sub-graphs, and change detection). These graphs can be constructed as a static graph, or a dynamic graph based on the availability of timestamp. Recent years have observed a

huge efforts on static graphs, among which Graph Convolutional Network (GCN) has appeared as a useful class of models. A challenge today is to detect anomalies with dynamic structures. In this chapter, we aim at providing methods used for detecting anomalies in static and dynamic graphs using graph analysis, graph embedding, and graph convolutional neural networks. For static graphs we categorize these methods according to plain and attribute static graphs. For dynamic graphs we categorize existing methods according to the type of anomalies that they can detect. Moreover, we focus on the challenges in this research area and discuss the strengths and weaknesses of various methods in each category. Finally, we provide open challenges for graph anomaly detection using graph convolutional neural networks on dynamic graphs.

Keywords: Anomaly Detection, Graph anomaly detection, Graph analysis, Graph embedding, Graph Neural Network, Dynamic graphs, Static graphs

1 Introduction

Anomaly detection is a critical task that deals with the problem of discovering "different from normal" signals or patterns by evaluating a large amount of data, thereby major flaws can be identified and avoided [2]. The objective of anomaly detection is to identify the data that are significantly different from most observations. Research on anomaly detection dates all the way back to the 1980s and detecting anomalies on graphs using machine learning has become an important research problem [2]. Graph anomaly detection has evolved from depending heavily on human experts' domain information into machine learning methods to eliminate human intervention. Recently, different deep learning approaches have been taken on to distinguish potential anomalies in graphs. These approaches are graph analysis [3], graph embedding [3], and graph neural networks [4]. By leveraging the graph structure, several of novel algorithms for anomaly detection have been proposed for each approach.

In recent years, graph neural networks (GNNs), as a powerful deep-learning-based graph representation method, have exhibited superiority in leveraging the graph structure and has been utilized in anomaly detection. Some researchers have successfully applied GNNs in several important anomaly detection tasks. Based on the availability of the timestamp, the graph can be constructed as a static graph or a dynamic graph, where a static graph refers to the graph that has fixed nodes and edges, and a dynamic graph refers to the graph that has nodes and/or edges change over time [2]. Dynamic graphs are more complex and difficult in discovering anomalies than static graphs. This can be shown from two perspectives: (1) The anomalous edges cannot be determined by the graph from a single timestamp. The detection procedure must take graph history into consideration; (2) Both the vertex and edge sets are changing over time [5]. As graph-based anomaly detection is becoming ever more important and the achievements of graph neural networks are

increasing, both academia and industry are interested in applying GNNs to handle the issue of anomaly detection on static and dynamic graphs. In this chapter, we summarize different GNN-based anomaly detection methods and provide taxonomies for them according to static and dynamic graphs. We will also see how GNN methods solved the limitations of graph analysis and graph embedding on both static and dynamic graphs.

The rest of this chapter is organized as follows. Section 2 identifies the approaches for analyzing graphs. Section 3 identifies the key challenges of Graph Neural Networks, and for Anomaly Detection on dynamic graphs. Section 4 presents methods for anomaly detection on static graphs and section 5 presents methods for anomaly detection on dynamic graphs. Section 6 identifies the requirements needed for better anomaly detection. In the last section, we provide the conclusion of this work.

2 Analyzing graphs

In this section we define the three core methods for analyzing graphs: graph analysis, graph embedding, and graph neural networks.

2.1 Graph Analysis

There is an increasing number of applications where data are represented as a graph with complex relationships and inter-dependency between objects. This means going from Euclidean (e.g., images, audio, and text) to non-Euclidean space representing interactions between nodes instead of the characteristics of individual points. Graph analysis [3] is a process for analyzing data in graph structures, in a **Non-Euclidean Space** using data points as nodes and relationships as edges. In order to perform the analysis on a non-Euclidean space, graph analysis methods have come into to improve the quantitative understanding and the control of complex networks.

2.2 Graph Embedding

Graph Embedding [3] methods have shown an important role for the capacity of transforming high-dimensional sparse graphs into **low-dimensional representations**, dense and continuous vector spaces. The main aim of graph embedding techniques is to encode nodes into a latent vector space and to group every node's property into a vector with a lower dimension.

2.3 Graph Neural Networks

Graph neural networks (GNNs) are classified into 4 categories: recurrent graph neural networks (RecGNNs), convolutional graph neural networks (ConvGNNs), graph autoencoders (GAEs), and spatial-temporal graph neural networks (STGNNs) [4]. In this chapter we will focus on Graph Convolutional Neural Networks (ConvGNNs) that acquire the movement of convolution from grid data (Euclidean structure) to graph data (non-Euclidean structure) [4].

ConvGNNs play an important role in building up many other complex GNN models [4]. They fall into two categories, spectral-based and spatial-based GNNs. Spectral based methods determine graph convolutions by proposing filters from graph signal processing [6]. Spatial-based methods achieve graph convolutions locally on each node where weights can be easily shared across different locations and structures [7].

Most of the current graph neural network methods have been modeled for static graphs, while many real-world graphs are being dynamic. Concurrently, designing graph neural networks for dynamic graphs is facing challenges. From the global perspective, structures of dynamic graphs remain evolving since new nodes and edges are always introduced. It is necessary to track the changing of graph neural network's structure. From the local perspective, a node can carry new edges, and the order of these edges is critical for learning the node [8]. In the next section we will identify the challenges for graph neural networks.

3 Key challenges

In this section we will first present the key challenges for graph neural networks and then identify the challenges for anomaly detection on dynamic graphs.

3.1 Challenges for Graph Neural Networks

While Graph Neural Networks have proved to be a very powerful approach for learning graph data, there are still several open challenges due to the complexity of graphs. Some of the challenges are listed below:

- **Model Depth**: [4] Deep learning model achievements depend on the building of neural networks. However, when using graphs, experimental studies have shown that with the increase in the number of layers, the model performance we be dropped dramatically [9]. This is caused by the effect of graph convolutions in pushing representations of adjacent nodes closer to each other. So it is noticed that with infinite number of convolutions, all nodes' representations will move to a single point. So, the challenge here is to evaluate whether going deep in the neural network layers can still improve graph data learning.
- **Scalability**:[4] At any time we try to scale or cluster our graph, it is an irrelevant problem that the completeness of the graph is sacrificed. The model will lose some part of graph information whether we are scaling or clustering. For scaling, a node can drop its significant neighbors. For clustering, a graph can drop a different structural pattern. Here comes the challenge of how it is possible to handle the scalability without sacrificing the integrity.
- **Heterogeneity**: [10] It is widely seen that most of GNN methods are applied on homogeneous graphs. The current GNN methods cannot handle the heterogeneous graph data, as it may have various types of nodes and edges or it may have different features. This calls for new approaches that can handle such heterogeneous data.

- **Dynamicity**: [10] Graphs have dynamic nature in a way that the nodes and edges keep changing. They can appear at some time and disappear at some other. Dynamic graphs are more complex due to the change of the graph structure. That is, the vertices and edges are unstable along the time dimension. Sometimes the nodes variate according to the time and the environment.

A challenge today is to deal with graphs with dynamic structures. It is important to study the anomaly detection on dynamic graphs using GNNs. The real-world networks can be modeled as dynamic graphs to represent the evolving objects and relationships among them. Apart from the structural information and node attributes, dynamic graphs also contain affluent temporal signals. On the one hand, this information inherently makes anomalous node detection on dynamic graphs more challenging. This is because dynamic graphs normally introduce massive volume of data and temporal signals must also be captured. On the other hand, they could provide more details about anomalies [8, 11, 12]. For instance, some anomalous nodes can appear to be normal in the graph snapshot at each time stamp, only when the graph structure variations are considered, they can be detected. In the next section we will provide the challenges for anomaly detection on dynamic graphs [1].

3.2 Challenges for Anomaly Detection on Dynamic graphs

Although anomaly detection has been an important research area for several years, there are still some unique and complex nature of anomalies which leads to unsolved challenges. Anomaly detection is not trivial due to its flexible and dynamic nature. Some anomalous operations display few explicit patterns but try to hide them in a large graph, while other operations are with constant patterns [13]. The core challenges for anomaly detection in dynamic graphs are the following ones:

- Some graphs are evolving over time which leads to new types of anomalies, for example, splitting, disappearing, or flickering communities [8].
- Graphs from many domains can exhibit entirely different behaviors over time. This divergence in evolution leads to application and approaches for specific anomalies [8].
- Some anomalies are slow to develop and span multiple time steps, thus it can be difficult to differentiate from organic graph evolution [8].
- The relationships between real objects and their inter-dependencies can no longer be treated individually for anomaly detection. The detection methods need to digest the deviating patterns of anomalies by considering the pairwise, triadic, and higher relationships among objects restored in conventional graphs [14, 15]. Moreover, the dynamic nature of real networks makes the detection problem much more challenging when the graph structure and attributes of nodes or edges change overtime [1].

4 Anomaly Detection on Static Graphs

In this section we define two types of static graphs: plain static, and attribute static graphs. We also define the methods used and the limitations of the three approaches: graph analysis, graph embedding, and graph convolutional neural networks in plain and attribute static graphs. Figure 1 shows how the methods are categorized.

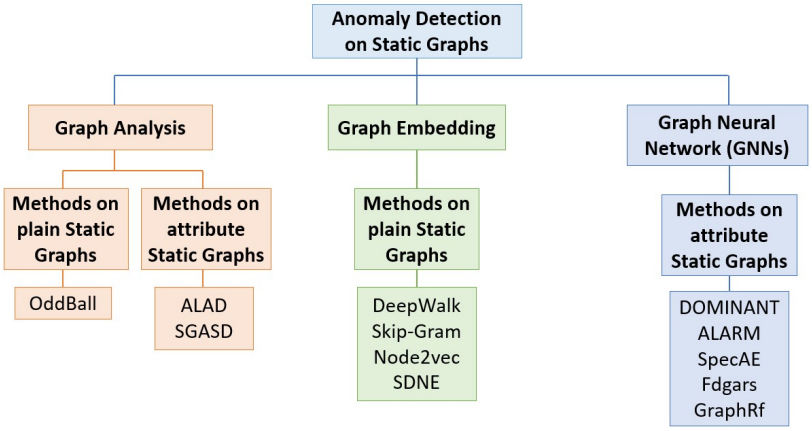

Fig. 1: Anomaly Detection methods applied on Static Graphs

4.1 Definitions

Plain Static Graph

A static plain graph $G = \{V, E\}$ contains a node set V and an edge set E. In a static plain graph, the graph structure consists of nodes $V = \{v_i\}$ and edges $E = \{e_{i,j}\}$ where n expresses the number of nodes and $e_{i,j} = (v_i, v_j)$ denotes an edge between nodes v_i and v_j. The $n \times n$ adjacency matrix $A = [a_{i,j}]$ restores the graph structure, where $a_{i,j} = 1$ if there exists an edge between nodes v_i and v_j , else $a_{i,j} = 0$ [1].

Attribute Static Graph

A static attributed graph $G = \{V, E, X\}$ contains a node set V, an edge set E and an attribute set X. In an attributed graph, the graph structure follows the previous definition. The $n \times k$ attribute matrix $X = [x_i]$ consists of nodes' attribute vectors, where x_i is the attribute vector associated with node v_i and k is the vector's dimension [1].

4.2 Anomaly detection on static graphs using Graph Analysis

Existing graph analysis methods for anomaly detection are established from machine learning models. The supervised learning methods learn the patterns of abnormal events and fit the non-linear models. These methods are reasonable for labeling the data as anomalies. The semi-supervised learning methods apply less labeled data to meet the models and detect the abnormal events on the unlabeled data. In general, these methods are constructed through an auto-encoder. If the loss induced by a data point is greater than the threshold, the data point is detected as an anomaly [16]. Normally these methods are not capable of detecting the anomaly without untrained patterns [16]. In this section we will divide the methods for anomaly detection using graph analysis into plain graphs and attribute graphs as follows:

4.2.1 Methods for plain static graphs

Plain graphs are dedicated to represent the structural information in real-world networks. For anomaly detection in plain graphs, the graph structure has been broadly exploited from different angles. A main idea behind that is to change the graph anomaly detection into a traditional anomaly detection issue, because the graph data with strong structure information cannot be handled by traditional detection methods. To bridge the gap, lots of works [17–19] manage to leverage the statistical features associated with each node such as in/out degree to detect anomalous nodes. For instance, OddBall [17] employs the statistical features (e.g., the number of 1-hop neighbors and edges, the total weight of edges) extracted from each node and its 1-hop neighbors to detect special structural anomalies that: 1) form local structures in shape of near-cliques or stars, 2) have heavy links with neighbors such that the total weight is extremely large, or 3) have a single dominant heavy link with one of the neighbors [1].

4.2.2 Limitations

When dealing with real word scenarios, it is very hard to choose the most suitable features from many candidates, and domain experts can always design new statistics, e.g., the maximum/minimum weight of edges. As a result, these methods lead to excessive costs for assessing the most significant features and cannot capture the structural information completely [1].

4.2.3 Methods for attribute static graphs

Apart from the structural information, real-world networks also contain affluent attribute information affiliated with nodes [20, 21]. These attributes supply complementary information over real objects and together with graph structure, more unseen anomalies that are non-trivial can now be discovered. Traditional techniques (e.g., statistical models, matrix factorization,

KNN) have been widely-applied to extract the structural/attribute patterns of anomalous nodes, after which the detection is performed. Among them, matrix factorization (MF) based techniques have shown power on capturing both the topological structure and node attributes. They achieve promising detection performance. For example, ALAD [22] measures the normality of each node according to attribute similarity with the community it belongs to. By ranking the nodes' normality scores in ascending order, the top-k nodes are identified as community anomalies. Linear regression models are also adopted to train anomaly classifiers given the labeled training data [1]. A representative work in [23] which is a supervised model, SGASD, that has yield encouraging results on identifying social spammers utilizing the social network structure, content information in social media and user labels [1].

4.2.4 Limitations

These graph analysis methods capture valuable information from the graph topology and node attributes, but their applications and generalizability's to real networks in large-scale are strictly limited due to the high computational cost of the matrix decomposition operation and regression models [1].

4.3 Anomaly detection on static graphs using graph embedding

In this section we present an overview of graph embedding methods that are used for anomaly detection on plain graphs.

4.3.1 Methods for plain static graphs

Graph embedding methods have been widely used to capture more valuable information from the graph structure for anomaly detection. Typically, these methods encode the graph structure into an embedded vector space and identify anomalous nodes through further analysis [1]. The first method of graph embedding is DeepWalk [11]. DeepWalk transforms graph structure information into sequences by random walk, and then Skip-Gram [24] is applied on it for each node embedding. After the success of DeepWalk, more graph embedding methods were designed. Node2vec [25] improves the random walk strategies of DeepWalk by a controllable deep or wide walking possibility. SDNE [26] introduces the Deep Auto-encoder model into graph embedding, modeling the 1st-order and 2ND-order neighbors of nodes. Using the graph embedding methods Skip gram, Node2vec, Deepwalk, both anomalous node and edge detection tasks can be applied with standard anomaly detection methods.

4.3.2 Limitations

Researchers easily use static graph embedding methods in every snapshot during different time steps, with constraint to align the same nodes in discrete time

steps. In this way, researchers claim that dynamic graph embedding methods could obtain better representations than traditional static graph embedding, since different snapshots share more characteristics for representation.

4.4 Anomaly detection on static graphs using graph convolutional neural network

In this section we present an overview of graph convolutional neural network methods that are used for anomaly detection on attribute graphs.

4.4.1 Methods for attribute static graphs

Graph convolutional neural networks (ConvGNNs) [27] have accomplished decent success in many graph data mining tasks (e.g., link prediction, node classification, and recommendation) owing to its capability of capturing comprehensive information in the graph structure and node attributes. Therefore, many anomalous node detection techniques start to investigate ConvGNNs. The proposed method, DOMINANT [28], comprises three parts, namely, graph convolutional encoder, structure reconstruction decoder, and attribute reconstruction decoder. The graph convolutional encoder generates node embeddings through multiple graph convolutional layers. For the purpose of capturing these signals, the proposed method, ALARM [29], applies multiple ConvGNNs to encode information in different views and adopts a weighted aggregation of them to generate node representations. The training strategy of this model is similar to DOMINANT [28] and aims at minimizing network reconstruction loss and attribute reconstruction loss. Lastly, ALARM adopts the same scoring function as [29], and nodes with top-k highest scores are anomalous. Instead of spotting unexpected nodes using their reconstruction errors, SpecAE [30] is used to detect global anomalies and community anomalies via a density estimation approach, Gaussian Mixture Model (GMM). The global anomalies can be identified by only considering the node attributes. For community anomalies, because of their distinctive attributes to the neighbors, the structure and attributes need to be jointly considered. Accordingly, SpecAE investigates a graph convolutional encoder to learn node representations and reconstruct the nodal attributes through a deconvolution decoder. In Fdgars[31] they developed a novel detection model that can identify fraudsters using their relations and features. This method firstly models online users' reviews and visited items as their features, and then identifies a small portion of significant fraudsters based on these features. In the last step, a ConvGNNs is trained in a semi-supervised manner by using the user network, user features, and labeled users. After training, the model can label unseen users directly. A more recent work, GraphRfi [32], also explores the potential of combining anomaly detection with other downstream graph analysis tasks. It targets on leveraging anomaly detection to identify malicious users and to provide more accurate recommendations to service benign users by alleviating the impact of these untrustworthy users.

4.4.2 Limitations

ConvGNNs has a simple convolution operation that aggregates neighbour information equally to the target node. This helps ConvGNNs to provide an effective solution to incorporate the graph structure with node attribute. However, ConvGNNs capability in capturing the most relevant information from neighbors is unsatisfactory.

5 Anomaly Detection on Dynamic Graphs

The real-world networks can be modeled as dynamic graphs to represent the evolving objects and relationships among them. In this section we define dynamic graph and present the type of anomalies in dynamic graph. We also present the methods used and the limitations of the three approaches: graph analysis, graph embedding, and graph convolutional neural network according to the different types of anomalies in dynamic graph.

5.1 Definitions

Dynamic Graphs

In dynamic graphs, vertices and edges can be inserted or removed at every time step. For simplicity, we assume that the vertex correspondence and the edge correspondence across different time steps are resolved because of unique labeling of vertices and edges, respectively. We define a graph series G as an ordered set of graphs with a fixed number of time steps. Formally, $\{G_t\}_{t=1}^{T}$ where T is the total number of time steps, $G_t = (V_t, E_t \subseteq (V_t \times V_t))$, and the vertex set V_t and edge set E_t may be plain or attributed (labeled). Graph series where $T \to \infty$ are called graph streams [1].

Types of anomalies on dynamic graphs

In this section, we identify and formalize four types of anomalies that arise in dynamic graphs.

1. **Anomalous Vertices** Anomalous vertex detection intends to find a subset of the vertices such that every vertex in the subset has an 'irregular' evolution compared to the other vertices in the graph [8]. Alternatively, the time points where the vertices are supposed to be anomalous can be detected. Dynamic graphs allow the temporal dynamics of the vertex to be involved, offering new types of anomalies that are not showed in static graphs. Anomalous vertices is formally defined as follows:
   Definition 1: Anomalous vertices (from [8]) Given G, the total vertex set $V = \bigcup_{t=1}^{T} V_t$, and a specified scoring function f : $V \to \mathbb{R}$, the set of anomalous vertices $V' \subseteq V$ is a vertex set such that $\forall\ v' \in V', |f(v') - \widehat{f}| > c_0$, where $\widehat{f}$ is a summary statistic of the scores $f(v), \forall\ v \in V$.
2. **Anomalous Edges** Similar to vertex detection, edge detection aims to find a subset of the edges such that every edge in the subset has an 'irregular'

evolution, optionally identifying the time points where they are abnormal [8]. Again, this concept can be generalized by assuming each method employs a function that maps each edge in the graph to a real number, low values indicating unusual behaviour. A high-level definition of anomalous edge detection can be defined as follows:

Definition 2: Anomalous edges (from [8]) Given G, the total edge set $E = \bigcup_{t=1}^{T} E_t$, and a specified scoring function $f : E \rightarrow \mathbb{R}$, the set of anomalous edges $E' \subseteq E$ is an edge set such that $\forall\ e' \in E'$, $|f(e') - \widehat{f}| > c_0$, where $\widehat{f}$ is a summary statistic of the scores $f(e), \forall\ e \in E$.

3. **Anomalous Subgraphs** Finding subgraphs with irregular behaviour requires an approach different from the ones for anomalous vertices or edges, as enumerating every possible subgraph in even a single graph is an intractable problem [8]. We define the anomalous subgraphs as follows:

   Definition 3: Anomalous subgraphs (from [8]) Given G, a subgraph set $H = U_{t=1}^{T} H_t$ where $H_t \subseteq G_t$ and a specified scoring function f : $H \rightarrow \mathbb{R}$, the set of anomalous subgraphs $H \subseteq H$ is a subgraph set such that $\forall\ h' \in H', |f(h') - \widehat{f}| > c_0$, where $\widehat{f}$ is a summary statistic of the scores $f(h), \forall\ h \in H$.

4. **Event and Change Detection** Event detection has attracted much interest in the data mining community, and it has a much broader scope compared to the previous three types of anomalies, aiming at identifying time points that are significantly different from the rest [8]. Isolated points in time where the graph is unlike the graphs at the previous and following time points represent events.

   Definition 4: Event detection (from [8]) Given G and a scoring function $f : G_t \rightarrow \mathbb{R}$, an event is defined as a time point t, such that $|f(G_t) - f(G_{t-1})| > c_0$ and $|f(G_t) - f(G_{t+1})| > c_0$. Now, we move on to the problem of change detection, which is complementary to event detection. It is important to note the distinction between event and change detection. While events represent isolated incidents, change points mark a point in time where the entire behaviour of the graph changes and the difference is maintained until the next change point [8].

   Definition 5: Change detection (from [8]) Given G and a scoring function $f : G_t \rightarrow \mathbb{R}$, a change is defined as a time point t, such that $|f(G_t) - f(G_{t-1})| > c_0$ and $|f(G_t) - f(G_{t+1})| \leqslant c_0$.

5.2 Anomaly detection on dynamic graphs using Graph Analysis

In this section we present an overview of graph analysis methods that are used for anomaly detection on dynamic graphs. These methods are divided according to three types of approaches: Vertex detection, subgraph detection, and change detection. Figure 2 shows how the methods are categorized. In addition, we present the limitations of these methods.

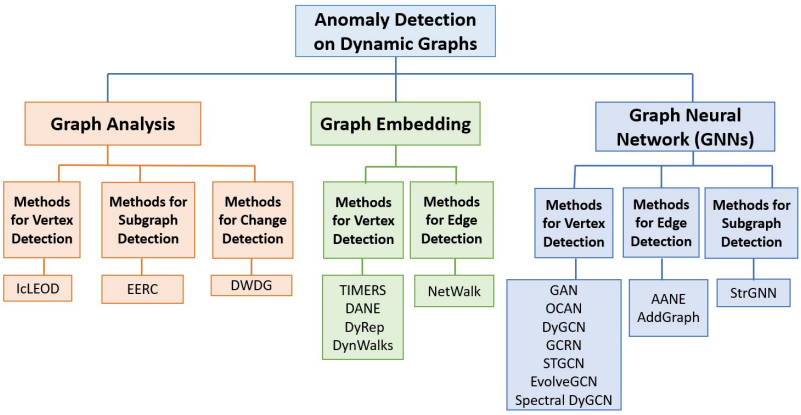

Fig. 2: Anomaly Detection methods applied on Dynamic Graphs

5.2.1 Methods for Vertex Detection

In [33] they proposed Incremental Local Evolutionary Outlier Detection (IcLEOD) method which illustrate that" group of vertices that belong to the same community are expected to exhibit similar behaviour. Probably this means that if at consecutive time steps one vertex in the community has a huge number of new edges added, the other vertices in the community would also have a huge number of new edges" [8]. If the remaining vertices in the community did not have new edges inserted, the vertex that did is detected as anomalous [34–37]. More regularly," a vertex's corenet consists of itself and all the vertices within two hops that have a weighted path above a threshold value" [8]." If the edge weight between two vertices is considered the strength of their connection, then intuitively the vertices connected with higher weight edges should be considered as a part of the same community" [8]." Consequently, if a vertex has two neighbors removed, one connected with a high edge weight and the other connected with a low edge weight, then the removal of the vertex connected by the higher edge weight should have more impact" [8]." At each time step, every vertex is first given an outlier score based on the change in its corenet, and the top-k outlier scores are then declared anomalous" [8].

5.2.2 Methods for Subgraph Detection

" Conversely, instead of finding changes, communities that are conserved, or stable, can be identified. Constructing multiple graphs at each time step based on different information sources, communities can be conserved across time and graphs" [8]. Graphs that act closely can be combined using clustering or prior knowledge. In [38]" the extreme events-related communities (EERC) method proposes that if a community is conserved across time steps and the graphs within its group, but has no corresponding community in any other group of graphs, then the community is defined as anomalous; two communities

are considered corresponding communities if they have a certain percentage of their vertices in common". Unlike [38–40] that consider only the structure of the graph, in social network there is often more knowledge available. A cluster that has an evolution event is considered as an anomalous subgraph once the evolution event occurs [8].

5.2.3 Methods for Change Detection

Changes are detected by partitioning the streaming graphs into coherent segments based on the similarity of their partitioning (communities) [8]. The start of each segment shows a detected change. The segments are found online by comparing the vertex partitioning of the newest graph to the partitioning found for the graphs in the current growing segment [8]. Vertex partitioning can be achieved with many methods, but in [36] they propose" the community mining including community discovery and change-point detection on dynamic weighted directed graphs (DWDG) that is achieved using the relevance matrix computed by random walks with restarts and modularity maximization". When the partitioning of the new graph is much different from the current segment's, a new segment begins, and the time point for the new graph is marked as a detected change [8]. The similarity of two partitions is computed as their Jaccard coefficient, and the similarity of two partitioning is the normalized sum of their partition similarities [8].

5.2.4 Limitations

The main problem here is that vertices and edges are changing along the time dimension, and we need to capture the dependency between different graphs along the time dimension [5].

5.3 Anomaly detection on dynamic graphs using graph embedding

In this section we present an overview of graph embedding methods that are used for anomaly detection on dynamic graphs. These methods are divided according to two types of anomalies vertex detection, and edge detection. In addition, we present the limitations of these methods.

5.3.1 Methods for Vertex Detection

Following the analysis of encoding graph into an embedding space, after which anomaly detection is applied, dynamic graph representation methods have been studied in the more recent works. TIMERS [41] proposes an incremental Singular Value Decomposition (SVD) model for dynamic embedding, which only needs SVD at beginning, and incrementally updates them according to graph change. DNE [42] proposes a dynamic version of LINE, with only a few gradient descent process to update the representation of current graph. DANE [43] proposes a graph embedding method in dynamic environment

which updates the node embedding based on the change in the adjacency matrix as well as in the attribute matrix via matrix perturbation. DyRep [44] ingests dynamic graph information in the form of association and communication events over time and updates the node representations as they appear in these events. DynWalks [45] incorporates the temporal information with traditional DeepWalk to capture the evolving properties in dynamic graphs. It updates nodes embeddings by sampling new walks which is highly related to the changes of graph [46]. Specifically, [47] introduces a flexible deep representation technique, NetWalk, to detect anomalous nodes in dynamic graphs using only the structure information [47]. It adopts an auto-encoder to learn node representations on the initial graph and incrementally updates them when new edges are added, or existing edges are deleted. For anomaly detection, NetWalk first adopts the streaming k-means clustering algorithm [48] to group existing nodes in the current time stamp into different clusters. Then, the anomaly score of each node is measured as its closest distance to the k clusters [48]. When node representations are updated, the cluster centers and anomaly scores are re-calculated accordingly.

5.3.2 Methods for Edge Detection

The intuition of graph representation-based techniques is to encode the dynamic graph structure information into edge representations and apply the traditional anomaly detection techniques to spot irregular edges. This is quite straightforward, but there remain vital challenges in generating/updating informative edge representations when the graph structure evolves. To mitigate this challenge, NetWalk [47] is also capable of detecting anomalous edges in dynamic graphs. Following the line of distance based anomaly detection, NetWalk encodes edges into a shared latent space using node embeddings and then detect anomalies based on their distances to the nearest edge-cluster centers in the latent space. When new edges arrive or existing edges disappear, the node and edge representations will be updated based on random walks in the temporary graphs at each time stamp, after which the edge-cluster centers and edge anomaly scores are re-calculated [47]. Finally, the top-k farthest edges to edge-clusters are reported as anomalies.

5.3.3 Limitations

Although NetWalk can detect anomalies in dynamic graphs, it simply updates edge representations without considering the long/short-term node and graph structure evolving patterns. NetWalk [47] approach the anomalous node detection problem promisingly, but they respectively only study the structure or attributes. Considering the graph embedding problem in a dynamic manner is more corresponding to the real-world applications. More information could be collected when we study the dynamic features of graph. Although some research on dynamic graph embedding have been proposed they mainly focus on mining the pattern of graph involvement, they still ignore the efficiency issue. Most of these works perform traditional static graph embedding in each

snapshot of a dynamic graph, and then consider adding constraint or interactions between different snapshots [46]. Their methods consequently get better performance than previous static methods. However, generating new representation at each time is costly, since most traditional node representation methods are supposed to learn the embedding parameters by optimization process (e.g., gradient descent or matrix factorization). Repeating this progress at each time step brings about high complexity [46].

5.4 Anomaly detection on dynamic graphs using graph convolutional neural network

In this section we present an overview of graph convolutional neural graph methods that are used for anomaly detection on dynamic graphs. Graph neural graphs have been applied to perform reasoning on the dynamics of physical systems. Graph convolutional neural networks, which extend the convolutional neural networks to graph structure data, have been shown to improve the performance of graph classification and vertex level semi-supervised classification. A general framework for graph neural networks is proposed in [8]. The dynamic information has been proven to boost a variety of graph analytical tasks such as community detection, link prediction and graph embedding. Therefore, it has strong potential to advance graph neural networks by considering the dynamic nature of graphs, which calls for dedicated efforts [8]. These methods are divided according to two types of anomalies: vertex detection and edge detection. In addition, we present the limitations of these methods.

5.4.1 Methods for Vertex Detection

Generative adversarial networks (GAN) [49] have received extensive attention because of their impressive performance in capturing real data distribution and generating simulated data. In [50], they circumvented the fraudster detection problem using only the observed benign users' attributes. The basic idea is to seize the normal activity patterns and detect anomalies which behave in a significant different manner. The proposed method, OCAN [50], starts with the extraction of benign users' content features using their historical social behaviors (e.g., historical posts, posts' URL), for which this method is classified into the dynamic category. Dynamic Graph Convolutional Network (DyGCN) [46], which is an extension of ConvGNNs-based methods. They naturally generalize the embedding propagation scheme of ConvGNNs to dynamic setting in an efficient manner, which is to propagate the change along the graph to update node embeddings. The most affected nodes are first updated, and then their changes are propagated to the next nodes and lead to their update. Extensive experiments conducted on various dynamic graphs demonstrate that their model can update the node embeddings in a time-saving and performance-preserving way [46]. The dynamic graph prediction task is different from dynamic graph embedding, but there are still some similarities. Graph

convolutional recurrent network (GCRN) [51] uses a Graph Convolutional Networks (ConvGNNs) to learn node embedding and feed it into the LSTM to learn dynamism. WD-ConvGNNs/CD-ConvGNNs [52] modifies the LSTM by incorporating it with ConvGNNs. Spatio-temporal Graph Convolutional Network (STGCN) [53] proposes a so-called ST-Conv blocks, which requires that the node features must be evolving over time. EvolveGCN [54] utilizes recurrent neural network (RNN) to evolve the ConvGNNs parameters, instead of the node embedding [46]. Spectral DyGCN [46] further upgrade the high-order mechanism above with acceptable time consumption. The high-order update mechanism ignores the global propagation of changing nodes. The changing information of 1-order nodes could not be propagated to all the nodes, which inevitably brings about the loss of accuracy.

5.4.2 Methods for Edge Detection

ConvGNNs can capture the structural information in edge detection. In AANE method [55] the authors demonstrate that the existence of anomalous edges in the training data prevents traditional ConvGNNs based models from capturing the real edge distribution, leading to sub-optimal detection performance. AANE solved the problem of detecting performance, by alleviating the negative impact of anomalous edges using the learned embeddings. AANE method iteratively updates the embeddings and detection results during training. In each training iteration, AANE generates node embeddings Z through ConvGNNs layers and learns an indicator matrix I to spot potential anomalous edges. AANE spots the top-k edges with lowest probabilities as anomalies [55]. Thus, an edge uv is identified as anomalous when its predicted probability is less than the average of all links associated with the node u by a predefined threshold. An alternative semi-supervised model, AddGraph, comprises a ConvGNNs and Gated Recurrent Units (GRU) with attention to capture more representative structural information from the temporal graph in each time stamp and the dependencies between them, respectively [13]. The hidden state of the nodes at each timestamp are used to calculate the anomalous probabilities of an existing edge and a negative sampled edge, and then feed them to a margin loss.

5.4.3 Methods for Subgraph Detection

In this section, we introduce a method for subgraph detection. Previous graph embedding based techniques have been frequently interesting on learning good node representations, whereas highly ignoring the sub-graph structural modifications related to the spot nodes in dynamic graphs. StrGNN [56] is an end-to-end structural temporal Graph Neural Network model for detecting anomalous edges in dynamic graphs. In precise, they first extract the hop enclosing subgraph centered on the spot edge and introduce the node labeling function to describe the role of each node in the [56]. Afterward, they drag graph convolution operation and sort pooling layer to extract the fixed-size feature from every snapshot/timestamp.

6 Open Challenges

Detecting anomalies using graph convolutional neural network is an important research direction, which leverages multi-source, multi-view features extracted from both content and structure for anomaly sample analysis and detection. It plays an important role in cyber security, but it still needs a lot of effort in the field due to the multiple issues from data, model and task. The future works are mainly lying in three perspectives: dynamic graphs, anomaly detection and graph machine learning.

Firstly, from dynamic graph learning perspective, there are two challenges [1]: Challenge 1 is the lack of raw attribute information on most dynamic graphs. Due to the explosive demand for data volume of time-evolving attributes or the inaccessible attribute evolved by privacy problem, it is difficult to construct attribute information to describe each node from the mainstream raw dynamic graph datasets. This leads to the need for an effective encoding method to represent evolving nodes. Challenge 2 is the difficulty of learning discriminative information from dynamic graphs where structural knowledge and temporal knowledge are coupled [1]. The idea is that both structural and temporal factors should be considered simultaneously when making the agreement, which raises a challenge in learning such coupled information.

Secondly, from an anomaly analysis perspective, there are still a lot of research questions [2]. How to define and identify the anomalies in the graph in the different tasks? How to effectively convert the large-scale raw data to the graph? How to effectively leverage the attributes? How to model the dynamic during the graph construction? How to keep the heterogeneity during the graph construction? Recently, due to data-specific and task-specific issues, the applications of GNN-based anomaly detection are still limited. There is still a lot of potential scenarios that can be applied.

Finally, from a graph machine learning perspective, lots of issues need to be addressed [2]. How to model the graph? How to represent the graph? How to leverage the context? How to fuse the content and structure features? Which part of the structure to capture, local or global? How to provide the model explainability? How to protect the model from adversarial attacks? How to overcome the time-space scalability bottleneck. Recently, lots of contributions have been made from the machine learning perspective. However, due to the unique characteristics of the anomaly detection problem, which GNNs to use and how to apply GNNs are still critical questions. Further work will also benefit from the new findings and new models in the graph machine learning community.

7 Conclusion

Due to the complex relationships between real-world objects and recent approaches for analyzing graph (especially graph convolutional neural networks), graph anomaly detection with deep learning is currently at the forefront of anomaly detection. In this chapter we review the methods for

graph anomaly detection with modern deep learning approaches. We divided these approaches between graph analysis, graph embedding and graph convolutional neural network. These approaches can be applied on static and dynamic graphs. For detecting anomalies on static graphs, we present graph analysis, graph embedding and graph convolutional neural network on plain static graphs and attribute static graphs. For detecting anomalies on dynamic graphs, we present graph analysis, graph embedding and graph convolutional neural network according to the types of graph anomalies they can detect as: (1) anomalous vertex detection; (2) anomalous edge detection; (3) anomalous subgraph detection and finally, (4) anomalous event and change detection. Clear summarizations and comparisons between different works are given, providing a complete and thorough picture of current work and the progress of graph anomaly detection as a field. Moreover, to push forward future research in this area, we provide open challenges for anomaly detection on dynamic graphs using graph neural networks.

References

[1] Ma, X., Wu, J., Xue, S., Yang, J., Zhou, C., Sheng, Q.Z., Xiong, H., Akoglu, L.: A comprehensive survey on graph anomaly detection with deep learning. IEEE Transactions on Knowledge and Data Engineering (2021)

[2] Shen Wang, P.S.Y.: Chapter 26 Graph Neural Networks in Anomaly Detection, pp. 558–578. https://graph-neural-networks.github.io/static/file/chapter26.pdf

[3] Xu, M.: Understanding graph embedding methods and their applications. arXiv preprint arXiv:2012.08019 (2020)

[4] Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., Philip, S.Y.: A comprehensive survey on graph neural networks. IEEE transactions on neural networks and learning systems **32**(1), 4–24 (2020)

[5] Zhou, R., Zhang, Q., Zhang, P., Niu, L., Lin, X.: Anomaly detection in dynamic attributed networks. Neural Computing and Applications **33**(6), 2125–2136 (2021)

[6] Shuman, D.I., Narang, S.K., Frossard, P., Ortega, A., Vandergheynst, P.: The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains. IEEE signal processing magazine **30**(3), 83–98 (2013)

[7] Guo, S., Lin, Y., Feng, N., Song, C., Wan, H.: Attention based spatial-temporal graph convolutional networks for traffic flow forecasting. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, pp.

922–929 (2019)

[8] Ranshous, S., Shen, S., Koutra, D., Harenberg, S., Faloutsos, C., Samatova, N.F.: Anomaly detection in dynamic networks: a survey. Wiley Interdisciplinary Reviews: Computational Statistics **7**(3), 223–247 (2015)

[9] Li, Q., Han, Z., Wu, X.-M.: Deeper insights into graph convolutional networks for semi-supervised learning. In: Thirty-Second AAAI Conference on Artificial Intelligence (2018)

[10] Gupta, A., Matta, P., Pant, B.: Graph neural network: Current state of art, challenges and applications. Materials Today: Proceedings (2021)

[11] Akoglu, L., Tong, H., Koutra, D.: Graph based anomaly detection and description: a survey. Data mining and knowledge discovery **29**(3), 626–688 (2015)

[12] Wang, H., Wu, J., Hu, W., Wu, X.: Detecting and assessing anomalous evolutionary behaviors of nodes in evolving social networks. ACM Transactions on Knowledge Discovery from Data (TKDD) **13**(1), 1–24 (2019)

[13] Zheng, L., Li, Z., Li, J., Li, Z., Gao, J.: Addgraph: Anomaly detection in dynamic graph using attention-based temporal gcn. In: IJCAI, pp. 4419–4425 (2019)

[14] Chen, H., Yin, H., Sun, X., Chen, T., Gabrys, B., Musial, K.: Multi-level graph convolutional networks for cross-platform anchor link prediction. In: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 1503–1511 (2020)

[15] Silva, J., Willett, R.: Hypergraph-based anomaly detection of high-dimensional co-occurrences. IEEE transactions on pattern analysis and machine intelligence **31**(3), 563–569 (2008)

[16] Li, G., Jung, J.J.: Entropy-based dynamic graph embedding for anomaly detection on multiple climate time series. Scientific Reports **11**(1), 1–10 (2021)

[17] Akoglu, L., McGlohon, M., Faloutsos, C.: Oddball: Spotting anomalies in weighted graphs. In: Pacific-Asia Conference on Knowledge Discovery and Data Mining, pp. 410–421 (2010). Springer

[18] Ding, Q., Katenka, N., Barford, P., Kolaczyk, E., Crovella, M.: Intrusion as (anti) social communication: characterization and detection. In: Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 886–894 (2012)

[19] Hooi, B., Song, H.A., Beutel, A., Shah, N., Shin, K., Faloutsos, C.: Fraudar: Bounding graph fraud in the face of camouflage. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 895–904 (2016)

[20] Hamilton, W.L., Ying, R., Leskovec, J.: Inductive representation learning on large graphs. In: Proceedings of the 31st International Conference on Neural Information Processing Systems, pp. 1025–1035 (2017)

[21] Hamilton, W.L., Ying, R., Leskovec, J.: Representation learning on graphs: Methods and applications. arXiv preprint arXiv:1709.05584 (2017)

[22] Liu, N., Huang, X., Hu, X.: Accelerated local anomaly detection via resolving attributed networks. In: IJCAI, pp. 2337–2343 (2017)

[23] Wu, L., Hu, X., Morstatter, F., Liu, H.: Adaptive spammer detection with sparse group modeling. In: Proceedings of the International AAAI Conference on Web and Social Media, vol. 11 (2017)

[24] Wang, Z., Lan, C.: Towards a hierarchical bayesian model of multi-view anomaly detection. In: IJCAI, pp. 2420–2426 (2020)

[25] Miao, K., Shi, X., Zhang, W.-A.: Attack signal estimation for intrusion detection in industrial control system. Computers & Security **96**, 101926 (2020)

[26] Jie, F., Wang, C., Chen, F., Li, L., Wu, X.: Block-structured optimization for anomalous pattern detection in interdependent networks. In: 2019 IEEE International Conference on Data Mining (ICDM), pp. 1138–1143 (2019). IEEE

[27] Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907 (2016)

[28] Ding, K., Li, J., Bhanushali, R., Liu, H.: Deep anomaly detection on attributed networks. In: Proceedings of the 2019 SIAM International Conference on Data Mining, pp. 594–602 (2019). SIAM

[29] Bandyopadhyay, S., Vivek, S.V., Murty, M.: Outlier resistant unsupervised deep architectures for attributed network embedding. In: Proceedings of the 13th International Conference on Web Search and Data Mining, pp. 25–33 (2020)

[30] Li, Y., Huang, X., Li, J., Du, M., Zou, N.: Specae: Spectral autoencoder for anomaly detection in attributed networks. In: Proceedings of the 28th ACM International Conference on Information and Knowledge

Management, pp. 2233–2236 (2019)

[31] Wang, J., Wen, R., Wu, C., Huang, Y., Xion, J.: Fdgars: Fraudster detection via graph convolutional networks in online app review system. In: Companion Proceedings of The 2019 World Wide Web Conference, pp. 310–316 (2019)

[32] Zhang, S., Yin, H., Chen, T., Hung, Q.V.N., Huang, Z., Cui, L.: Gcn-based user representation learning for unifying robust recommendation and fraudster detection. In: Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 689–698 (2020)

[33] Ji, T., Yang, D., Gao, J.: Incremental local evolutionary outlier detection for dynamic social networks. In: Joint European Conference on Machine Learning and Knowledge Discovery in Databases, pp. 1–15 (2013). Springer

[34] Saligrama, V., Chen, Z.: Video anomaly detection based on local statistical aggregates. In: 2012 IEEE Conference on Computer Vision and Pattern Recognition, pp. 2112–2119 (2012). IEEE

[35] Tran, L., Navasca, C., Luo, J.: Video detection anomaly via low-rank and sparse decompositions. In: 2012 Western New York Image Processing Workshop, pp. 17–20 (2012). IEEE

[36] Duan, D., Li, Y., Jin, Y., Lu, Z.: Community mining on dynamic weighted directed graphs. In: Proceedings of the 1st ACM International Workshop on Complex Networks Meet Information & Knowledge Management, pp. 11–18 (2009)

[37] Tantipathananandh, C., Berger-Wolf, T.Y.: Finding communities in dynamic social networks. In: 2011 IEEE 11th International Conference on Data Mining, pp. 1236–1241 (2011). IEEE

[38] Chen, Z., Hendrix, W., Guan, H., Tetteh, I.K., Choudhary, A., Semazzi, F., Samatova, N.F.: Discovery of extreme events-related communities in contrasting groups of physical system networks. Data Mining and Knowledge Discovery **27**(2), 225–258 (2013)

[39] Chen, Z., Hendrix, W., Samatova, N.F.: Community-based anomaly detection in evolutionary networks. Journal of Intelligent Information Systems **39**(1), 59–85 (2012)

[40] Araujo, M., Papadimitriou, S., Günnemann, S., Faloutsos, C., Basu, P., Swami, A., Papalexakis, E.E., Koutra, D.: Com2: fast automatic discovery of temporal ('comet') communities. In: Pacific-Asia Conference on

Knowledge Discovery and Data Mining, pp. 271–283 (2014). Springer

[41] Zhang, Z., Cui, P., Pei, J., Wang, X., Zhu, W.: Timers: Error-bounded svd restart on dynamic networks. In: Thirty-Second AAAI Conference on Artificial Intelligence (2018)

[42] Du, L., Wang, Y., Song, G., Lu, Z., Wang, J.: Dynamic network embedding: An extended approach for skip-gram based network embedding. In: IJCAI, vol. 2018, pp. 2086–2092 (2018)

[43] Li, J., Dani, H., Hu, X., Tang, J., Chang, Y., Liu, H.: Attributed network embedding for learning in a dynamic environment. In: Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, pp. 387–396 (2017)

[44] Trivedi, R., Farajtabar, M., Biswal, P., Zha, H.: Dyrep: Learning representations over dynamic graphs. In: International Conference on Learning Representations (2019)

[45] Hou, C., Zhang, H., Tang, K., He, S.: Dynwalks: Global topology and recent changes awareness dynamic network embedding. arXiv preprint arXiv:1907.11968 (2019)

[46] Cui, Z., Li, Z., Wu, S., Zhang, X., Liu, Q., Wang, L., Ai, M.: Dygcn: Dynamic graph embedding with graph convolutional network. arXiv preprint arXiv:2104.02962 (2021)

[47] Yu, W., Cheng, W., Aggarwal, C.C., Zhang, K., Chen, H., Wang, W.: Netwalk: A flexible deep embedding approach for anomaly detection in dynamic networks. In: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 2672–2681 (2018)

[48] Zhang, Y., Tangwongsan, K., Tirthapura, S.: Streaming k-means clustering with fast queries. In: 2017 IEEE 33rd International Conference on Data Engineering (ICDE), pp. 449–460 (2017). IEEE

[49] Liu, Z., Dou, Y., Yu, P.S., Deng, Y., Peng, H.: Alleviating the inconsistency problem of applying graph neural network to fraud detection. In: Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 1569–1572 (2020)

[50] Zheng, P., Yuan, S., Wu, X., Li, J., Lu, A.: One-class adversarial nets for fraud detection. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, pp. 1286–1293 (2019)

[51] Seo, Y., Defferrard, M., Vandergheynst, P., Bresson, X.: Structured sequence modeling with graph convolutional recurrent networks. In: International Conference on Neural Information Processing, pp. 362–373 (2018). Springer

[52] Manessi, F., Rozza, A., Manzo, M.: Dynamic graph convolutional networks. Pattern Recognition **97**, 107000 (2020)

[53] Xu, K., Hu, W., Leskovec, J., Jegelka, S.: How powerful are graph neural networks? arXiv preprint arXiv:1810.00826 (2018)

[54] Pareja, A., Domeniconi, G., Chen, J., Ma, T., Suzumura, T., Kanezashi, H., Kaler, T., Schardl, T., Leiserson, C.: Evolvegcn: Evolving graph convolutional networks for dynamic graphs. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, pp. 5363–5370 (2020)

[55] Duan, D., Tong, L., Li, Y., Lu, J., Shi, L., Zhang, C.: Aane: Anomaly aware network embedding for anomalous link detection. In: 2020 IEEE International Conference on Data Mining (ICDM), pp. 1002–1007 (2020). IEEE

[56] Cai, L., Chen, Z., Luo, C., Gui, J., Ni, J., Li, D., Chen, H.: Structural temporal graph neural networks for anomaly detection in dynamic graphs. arXiv preprint arXiv:2005.07427 (2020)