A Comprehensive Survey on Graph Anomaly Detection with Deep Learning

Xiaoxiao Ma, Jia Wu, *Senior Member, IEEE,* Shan Xue, Jian Yang, Chuan Zhou
Quan Z. Sheng, and Hui Xiong, *Fellow, IEEE* and Leman Akoglu

Abstract—Anomalies represent rare observations (e.g., data records or events) that deviate significantly from others. Over several decades, research on anomaly mining has received increasing interests and the burst of information has attracted more attention on anomalies because of their significance in a wide range of disciplines (e.g., security, finance, medicine). Anomaly detection, which aims to identify these rare observations, is among the most vital tasks in the world, and has shown its power in preventing detrimental events, such as financial fraud, network intrusion, and social spam. The detection task is typically solved by identifying outlying data points in the feature space and inherently overlooks the relational information in real-world data. Graphs have been prevalently used to represent the structural/relational information, which raises the *graph anomaly detection problem* - identifying anomalous graph objects (i.e., nodes, edges and sub-graphs) in a single graph, or anomalous graphs in a sequence/set/database of graphs. However, conventional anomaly detection techniques cannot tackle 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.). Thanks to the advent of deep learning in breaking these limitations, graph anomaly detection with deep learning has received a growing attention recently. In this survey, we aim to provide a systematic and comprehensive review of the contemporary deep learning techniques for graph anomaly detection. Specifically, we provide a taxonomy that follows a task-driven strategy and categorizes existing work according to the anomalous graph objects that they can detect. We especially focus on the challenges in this research area and discuss the key intuitions, technical details as well as relative strengths and weaknesses of various techniques in each category. Moreover, we compile open-sourced implementations, public datasets, and commonly-used evaluation metrics to provide affluent resources for future studies. Finally, we highlight twelve extensive future research directions according to our survey results covering unsolved and emerging problems introduced by graph data, anomaly detection, deep learning and real-world applications. With this survey, our goal is to create a "one-stop-shop" that provides a unified understanding of the problem categories and existing approaches, publicly available hands-on resources, and high-impact open challenges for graph anomaly detection using deep learning.

Index Terms—Anomaly Detection, Graph Anomaly Detection, Deep Learning, Graph Mining, Graph Neural Networks.

✦

1 Introduction

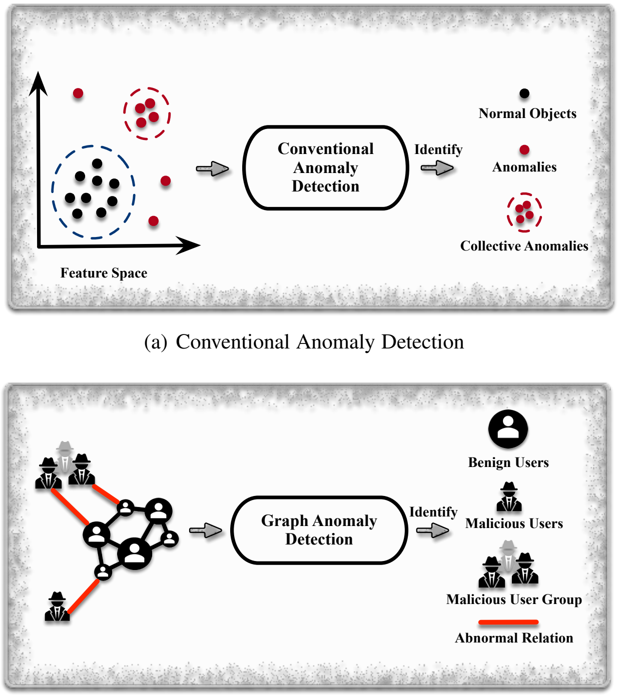
An anomaly or an outlier is first defined by Grubbs [1] as *"one that appears to deviate markedly from other members of the sample in which it occurs"* in 1969 and the studies on anomaly detection is initiated by the statistics community in the 19th century. In most cases, anomalies might appear as social spammers or misinformation in social media; fraudsters, bot users or sexual predators in social networks; network intruders or malware in computer networks and broken devices or malfunctioning blocks in industry systems, and they often introduce huge damage the real-world systems they appear in. According to FBI's 2014 Internet Crime Report[1], the financial loss of crimes on social media reached more than $60 million in the second half year and a more up-to-date report on site[2] indicates that the global economic cost of online fake news reaches around $78 billion a year in 2020.

In computer science, the research on anomaly detection dates back to the 1980s and detecting anomalies on graph data has emerged as an important data mining paradigm since then. The extensive presence of connections between real-world objects and advances in graph data mining in the last decade have revolutionized our understanding of the graph anomaly detection problems and this research field has received a dramatic increase in interest in the past 5 years. One of the most significant changes is that graph anomaly detection has evolved from relying heavily on human experts' domain knowledge into machine learning techniques to eliminate human intervention, and more recently, various deep learning technologies have been adopted to identify potential anomalies in graphs more accurately and in real-time. An increasing number of contemporary deep learning based graph anomaly detection techniques have been deployed to many real applications, including: financial fraud detection, social spam detection, network intrusion detection, misinformation detection, industrial system damage detection, etc., and achieved success in reducing the damage of anomalies. As a frontier technology, graph anomaly detection with deep learning, hence, is expected to generate more fruitful results on detecting anomalies and secure a

- *X. Ma, J. Wu, S. Xue, J. Yang, Q. Z. Sheng are with Department of Computing, Macquarie University, Sydney, NSW 2109, Australia. E-mail: {xiaoxiao.ma2@students., jia.wu, emma.xue, jian.yang, michael.sheng}mq.edu.au.*
- *Chuan Zhou is with the Academy of Mathematics and Systems Science, Chinese Academy of Sciences, Beijing, China. E-mail: zhouchuan@amss.ac.cn.*
- *H. Xiong is with Department of Management Science and Information Systems, Rutgers, the State University of New Jersey, USA. Email: hxiong@rutgers.edu.*
- *Leman Akoglu is the Heinz College Dean's Associate Professor at Carnegie Mellon University's Heinz College of Information Systems and Public Policy, USA. Email: lakoglu@andrew.cmu.edu.*

[1] https://www.fbi.gov/file-repository/2014_ic3report.pdf/view

[2] https://www.zdnet.com/article/online-fake-news-costing-us-78-billion-globally-each-year/



(a) Conventional Anomaly Detection

(b) Graph Anomaly Detection

Fig. 1: Toy Examples of Conventional Anomaly Detection and Graph Anomaly Detection. Apart from anomalies shown in (b), graph anomaly detection also identifies graph-level anomalies, detailed in Section 8.

more convenient life for the society.

Anomalies, which are also known as outliers, exceptions, peculiarities, rarities, novelties, etc., in different application fields, refer to abnormal objects that are significantly different from the standard, normal, or expected. For instance, spammers in social networks, fake news in social media and abnormal network traffic in computer networks are well-known anomalies in our daily life. Although these objects rarely occur in real-world, they contain critical information to support downstream applications. For example, the behaviours of fraudsters provide evidences for anti-fraud detection and abnormal network traffics reveal signals for network intrusion protection. Anomalies, in many cases, may also have a range of adverse impacts, for instance, fake news in social media would create panic and chaos in society and mislead the beliefs of the public [2]–[5], untrustworthy reviews in online review systems affect customers' shopping choices [6]–[8], network intrusions might leak private personal information to hackers [9]–[12], and financial frauds would cause huge damage to economic systems [13]–[16]. Indeed, anomalies have received great attention by researchers in different disciplines and due to their increasing (negative) impact, there is an increasing demand for detecting potential anomalies in a wide-range of real-world applications recently.

Anomaly detection is the data mining process that aims to identify the unusual patterns that deviate from the majorities in a dataset [17]–[19]. The anomalies may appear in the form of abnormal data records, messages, events, groups, and/or other unexpected observations. In order to detect anomalies, conventional techniques typically represent real-world objects as feature vectors (e.g., news in social media are represented as bag-of-words [20], and images in web pages are represented as color histograms [21]), and then detect outlying data points in the vector space [22]–[24], as shown in Fig. 1(a). Although these techniques have shown power in locating deviating data points under tabulated data format, they inherently discard the complex relationships between objects [25].

In reality, many objects have rich relationships with others, which can provide valuable complementary information for anomaly detection. Take online social networks as an example, fake users can be created using valid information from normal users or they can camouflage themselves by mimicking benign users' attributes [26], [27]. Hence, fake users and benign users would have near-identical features, and conventional anomaly detection techniques would be less capable of identifying them using the feature information only. Meanwhile, fake users always build relationships with a large number of benign users to increase their reputation and influence so they can get unexpected benefits, but benign users rarely exhibit such activities [28], [29]. Hence, these dense and unexpected connections formed by fake users denote their deviations to the benigns and more comprehensive detection techniques should take these structural information into account to pinpoint the deviating patterns of anomalies.

To represent the structural information, *Graphs* have been prevalently used in many application fields [30]–[35], such as social activities, e-commerce, biology, academy and communication. Specifically, in graphs, the nodes/vertices represent real objects and the edges represent their relationships. With the structural information comprised in graphs, detecting anomalies in graphs raises a more complex anomaly detection problem in non-Euclidean space - graph anomaly detection (GAD) that aims to identify anomalous graph objects (i.e., nodes, edges or sub-graphs) in a single graph as well as anomalous graphs among a set/database of graphs [25], [36], [37]. As a toy example shown in Fig. 1(b), given an online social network, graph anomaly detection aims to identify anomalous nodes (i.e., malicious users), anomalous edges (i.e., abnormal relations) and anomalous sub-graphs (i.e., malicious user groups), respectively. As a result, traditional anomaly detection techniques are unfeasible to be directly applied for graph anomaly detection because the copious types of graph anomalies that cannot be directly represented in the Euclidean feature space, and researchers have intensified their efforts to detect anomalies on graphs recently.

In fact, a handful research works have been done for graph anomaly detection. Amongst earlier works in this area, the detection methods rely heavily on handcrafted feature engineering or statistical models built by domain experts [38]–[40]. This inherently limits these techniques' capability to detect unknown anomalies and it is very human labor intensive. Many machine learning techniques, such as matrix factorization [41] and SVM [42], have also been applied to detect graph anomalies. However, real-world networks often contain millions of nodes and edges that result in extremely high dimensional and large-scale data, and these techniques do not easily scale up to such data efficiently. Practically, they exhibit high computational overhead in both the storage and execution time [43]. These general challenges associated with graph data are significant for the detection techniques, and we categorize them as data-specific challenges (Data-CHs) in this survey. A summary of them is provided in Appendix A.

In the meantime, the non-deep learning based techniques also lack the capability to capture the non-linear properties of real objects [24], hence the representations of objects learned by them are not expressive enough to fully support graph anomaly

TABLE 1: A Comparison Between Existing Surveys on Anomaly Detection. We mark edge and sub-graph detection as ● in our survey because we review more deep learning based works than any previous surveys.

Surveys	AD	DAD	GAD	GADL				Source Code	Dataset	
				Node	Edge	Sub-graph	Graph		Real-world	Synthetic
Our Survey	●	●	●	●	●	●	●	●	●	●
Chandola et al. [18]	●	-	-	-	-	-	-	-	-	-
Boukerche et al. [44]	●	◐	-	-	-	-	-	-	-	-
Bulusu et al. [45]	●	●	-	-	-	-	-	-	-	-
Thudumu et al. [43]	●	○	○	-	-	-	-	-	-	-
Pang et al. [24]	●	●	○	○	○	-	-	●	●	-
Chalapathy and Chawla [46]	●	●	-	-	-	-	-	●	●	-
Akoglu et al. [25]	●	-	●	-	-	-	-	-	-	-
Ranshous et al. [47]	●	-	●	-	-	-	-	●	-	-
Jennifer and Kumar [48]	●	-	●	-	-	-	-	-	-	-
Eltanbouly et al. [49]	●	◐	○	-	-	-	-	-	-	-
Fernandes et al. [50]	●	◐	●	-	-	-	-	-	-	-
Kwon et al. [51]	●	◐	-	-	-	-	-	-	○	-
Gogoi et al. [52]	●	○	-	-	-	-	-	-	-	-
Savage et al. [53]	●	-	●	-	-	-	-	-	-	-
Yu et al. [6]	●	-	●	-	-	-	-	-	-	-
Hunkelmann et al. [3]	●	-	○	-	-	-	-	-	-	-
Pourhabibi et al. [19]	●	◐	●	◐	-	-	-	-	-	-

* AD: Anomaly Detection, DAD: Anomaly Detection with Deep Learning, GAD: Graph Anomaly Detection.

* GADL: Graph Anomaly Detection with Deep Learning.

* -: not included, ○ (1-2 references included), ◐ (3-10 references included), ● (10+ references included).

detection. In order to tackle these problems, more recent studies seek the potential of adopting deep learning techniques to identify anomalous graph objects. As a powerful tool for data mining, deep learning has achieved great success in data representation and pattern recognition [54]–[56] as a deep architecture with layers of parameters and transformations appear to suit the aforementioned problems well, which conventional machine learning techniques met in practice. The more recent studies, such as deep graph representation learning and graph neural networks (GNNs), further enrich the capability of deep learning for graph data mining [57]–[61]. By extracting expressive representations such that graph anomalies and normal objects can be easily separated, or learning the deviating patterns of anomalies directly through deep learning techniques, graph anomaly detection with deep learning (GADL) is starting to take the lead in the forefront of anomaly detection.

1.1 Challenges in GAD with Deep Learning

Due to the complexity of anomaly detection and graph data mining [62]–[66], in addition to the prior mentioned data-specific challenges, adopting deep learning techniques for graph anomaly detection also faces a number of challenges from the technical side. Specifically, we categorize these challenges associated with deep learning as technique-specific challenges (Tech-CHs) and summarize them as follows.

Tech-CH1. Anomaly-aware training objectives. Deep learning models rely heavily on the training objectives to fine-tune all the trainable parameters. For graph anomaly detection, this necessitates appropriate training objectives or loss functions such that the GADL models can effectively capture the differences between benign and anomalous objects. Designing anomaly-aware objectives is very challenging because there is no prior knowledge about the ground-truth anomalies as well as their deviating patterns to the majority. How to effectively separate anomalies from normal objects through training remains critical for deep learning-based models.

Tech-CH2. Anomaly interpretability. In real-world scenarios, the interpretability of detected anomalies is also vital because we need to provide convincing evidence to support the subsequent anomaly handling process. For example, the system and risk management department of financial organizations must provide lawful evidence before blocking the account of identified anomalous users. As deep learning has been limited for its interpretability [24], [67], how to explain the detected graph anomalies remains a big challenge for deep learning techniques.

Tech-CH3. High training cost. Although D(G)NNs are capable of digesting affluent information (e.g., structural information and attributes) in graph data for anomaly detection, these GADL models are more complex than conventional deep neural networks or machine learning methods due to the anomaly-aware training objectives. Such complexity inherently leads to high training costs on both time and computing resources.

Tech-CH4. Hyperparameter tuning. D(G)NNs naturally exhibit a large set of hyperparameters, such as the number of neurons in each neural network layer, learning rate, weight decay and the number of training epochs. Their learning performance is significantly affected by the values of these hyperparameters. However, it remains a serious challenge to effectively selecting the optimal/sub-optimal settings for the detection models due to the unavailability of labeled data in real scenarios.

Because deep learning models are sensitive to their associated hyperparameters, setting well-performing values for the hyperparameters becomes vital vital to real-applications. Tuning hyperparameter is relatively trivial in supervised learning when labeled data are available. Users could find an optimal/sub-optimal set of hyperparameters (e.g., through random search, grid search) by comparing the model's outputs with the ground-truth. However, unsupervised anomaly detection has no accessible labeled data to judge the model's performance under different hyperparameter settings [68], [69]. Selecting the ideal hyperparameter values for unsupervised detection models persists as a critical obstacle to applying them in a wide range of real-scenarios.

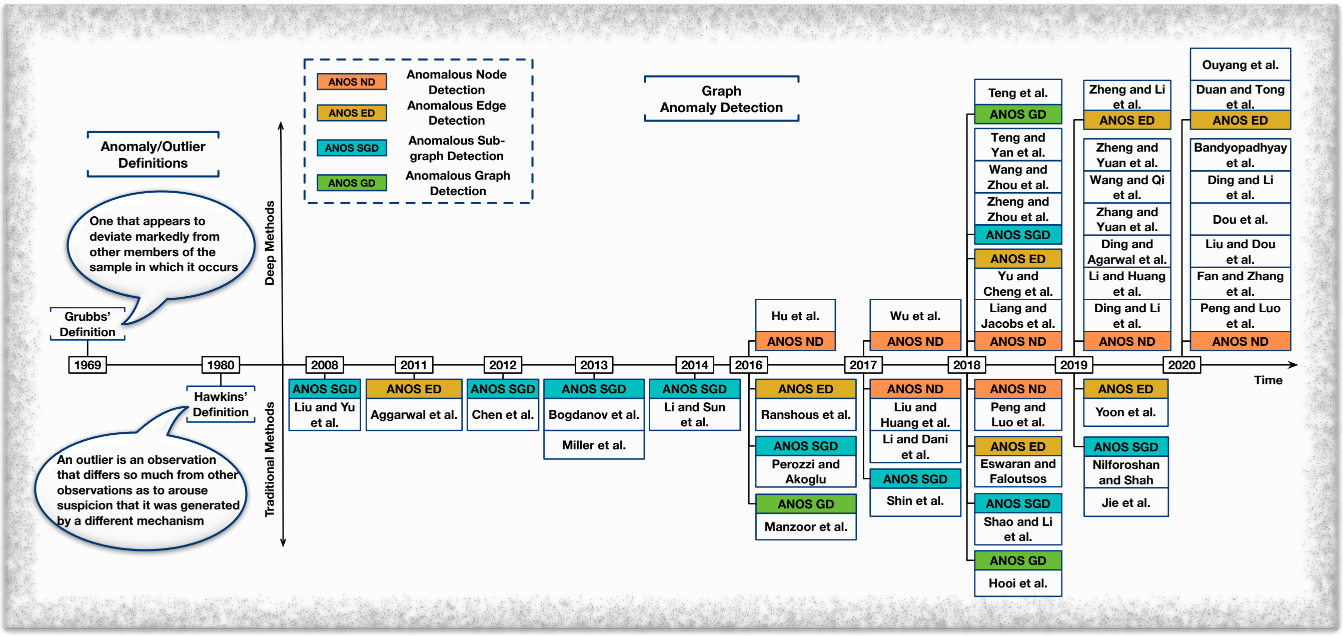

ANOS ND
Anomalous Node Detection
ANOS ED
Anomalous Edge Detection
ANOS SGD
Anomalous Sub-graph Detection
ANOS GD
Anomalous Graph Detection
Graph Anomaly Detection
Anomaly/Outlier Definitions
One that appears to deviate markedly from other members of the sample in which it occurs
Grubbs' Definition
Hawkins' Definition
An outlier is an observation that differs so much from other observations as to arouse suspicion that it was generated by a different mechanism
Deep Methods
Traditional Methods
1969
1980
2008
2011
2012
2013
2014
2016
2017
2018
2019
2020
Time
ANOS SGD Liu and Yu et al.
ANOS ED Aggarwal et al.
ANOS SGD Chen et al.
ANOS SGD Bogdanov et al.
Miller et al.
ANOS SGD Li and Sun et al.
Hu et al. ANOS ND
ANOS ED Ranshous et al.
ANOS SGD Perozzi and Akoglu
ANOS GD Manzoor et al.
Wu et al. ANOS ND
ANOS ND Liu and Huang et al.
Li and Dani et al.
ANOS SGD Shin et al.
Teng et al. ANOS GD
Teng and Yan et al.
Wang and Zhou et al.
Zheng and Zhou et al. ANOS SGD
ANOS ED Yu and Cheng et al.
Liang and Jacobs et al. ANOS ND
ANOS ND Peng and Luo et al.
ANOS ED Eswaran and Faloutsos
ANOS SGD Shao and Li et al.
ANOS GD Hooi et al.
Zheng and Li et al. ANOS ED
Zheng and Yuan et al.
Wang and Qi et al.
Zhang and Yuan et al.
Ding and Agarwal et al.
Li and Huang et al.
Ding and Li et al. ANOS ND
ANOS ED Yoon et al.
ANOS SGD Nilforoshan and Shah
Jie et al.
Ouyang et al.
Duan and Tong et al. ANOS ED
Bandyopadhyay et al.
Ding and Li et al.
Dou et al.
Liu and Dou et al.
Fan and Zhang et al.
Peng and Luo et al. ANOS ND

Fig. 2: A Timeline of Graph Anomaly Detection and Reviewed Techniques.

1.2 Existing Anomaly Detection Surveys

Recognizing the significance of anomaly detection, many review works have been conducted in the last ten years covering the topics of anomaly detection, anomaly detection with deep learning, graph anomaly detection, graph anomaly detection with deep learning, and particular applications of graph anomaly detection (i.e., social media, social networks, fraud detection and network security).

Specifically, [18], [44] and [43] are representative surveys on the generalized anomaly detection techniques. But only the most up-to-date work in Thudumu et al. [43] covers the topic of graph anomaly detection. Recognizing the power of deep learning, the three contemporary surveys, Ruff et al. [70], Pang et al. [24] and Chalapathy and Chawla [46] reviewed deep learning based anomaly detection techniques specifically.

As for graph anomaly detection, Akoglu et al. [25], Ranshous et al. [47], and Jennifer and Kumar [48] put their concentration on graph anomaly detection and reviewed many conventional approaches in this area including statistical models and machine learning techniques. Other surveys (e.g., [3], [6], [19], and [49]–[53].) were dedicated to particular applications of graph anomaly detection, such as computer network intrusion detection and anomaly detection in online social networks. These works provided solid reviews toward the application of anomaly detection/graph anomaly detection techniques in these high demand and vital domains. However, none of the mentioned surveys are dedicated to techniques on graph anomaly detection with deep learning as shown in Table 1, and hence do not provide a systematic and comprehensive review of these techniques.

1.3 Contributions

Our contributions are summarized as follows:

- **The first survey in graph anomaly detection with deep learning.** To the best of our knowledge, our survey is the first to review the-state-of-the-art deep learning techniques for graph anomaly detection. Most of the relevant surveys focus either on conventional graph anomaly detection methods using non-deep learning techniques or on generalized anomaly detection techniques (for tabular/point data, time series, etc.). There is no dedicated and comprehensive survey on graph anomaly detection with deep learning until now. Our work bridges the gap and we expect that an organized and systematic survey will help push forward the research in this area.
- **A systematic and comprehensive review.** In this survey, we review the most up-to-date deep learning techniques for graph anomaly detection published in influential international conferences and journals in the area of deep learning, data mining, web services, and artificial intelligence, including: TKDE, TKDD, TPAMI, NeurIPS, SIGKDD, ICDM, WSDM, SDM, SIGMOD, IJCAI, AAAI, ICDE, CIKM, ICML, WWW, CVPR, and others. We first summarize seven data-specific and four technique-specific challenges in graph anomaly detection with deep learning, respectively, and then review existing works comprehensively from the perspectives of: 1) motivations of deep methods, 2) main ideas to identify graph anomalies, 3) brief introduction to conventional non-deep learning techniques, and 4) technical details of deep learning algorithms. A brief timeline of graph anomaly detection and reviewed works is given in Fig. 2.
- **Future directions.** According to our survey results, we highlight twelve future research directions covering emerging problems introduced by graph data, anomaly detection, deep learning models, and real-world applications. These future opportunities indicate challenges that have not been adequately tackled and more efforts are needed in the future.
- **Affluent resources.** Our survey also provides an extensive collection of open-sourced anomaly detection algorithms, public datasets, synthetic dataset generating techniques, as well as commonly used evaluation metrics to push forward the state of the art in graph anomaly detection. These published resources offer benchmark datasets and

baselines for future research.

- **A new taxonomy.** We organize this survey with regard to different types of anomalies (i.e., nodes, edges, sub-graphs, and graphs) existing in graphs or graph databases. We also pinpoint the differences and similarities between different types of graph anomalies.

The rest of this survey is organized as follows. In Section 2, we provide preliminaries about different types of settings. From Section 3 to Section 8, we review existing anomalous node, edge, sub-graph and graph detection techniques, respectively. In Section 9, we first provide a collection of published graph anomaly detection algorithms and datasets, and then summarize commonly used evaluation metrics and synthetic data generation strategies. We highlight twelve future directions concerning deep learning in graph anomaly detection in Section 10 and summarize our survey in Section 11. A concrete taxonomy of our survey is shown in Appendix B.

2 Preliminaries

In this section, we provide definitions of different types of graphs mostly used in node/edge/sub-graph-level anomaly detection (Section 3 to Section 7). For consistency, we follow the conventional categorization of graphs as in existing works [25], [47], [51] and categorize them as static, dynamic graphs, and graph database. Unless specified, all graphs mentioned in the following sections are static. Meanwhile, as graph-level anomaly detection is discussed far away from page 13, to enhance readability, we provide the related definition, graph database in Section 8.

Definition 1 (Plain Graph). A static plain graph $G = \{V, E\}$ consists of a node set V and an edge set E. In a static plain graph, the graph structure is formed by nodes $V = \{v_i\}_1^n$ and edges $E = \{e_{i,j}\}$ where n denotes the number of nodes and $e_{i,j} = (v_i, v_j)$ represents an edge between nodes v_i and v_j. The adjacency matrix $A = [a_{i,j}]_{n\times n}$ restores the graph structure, where $a_{i,j} = 1$ if there exists an edge between node v_i and v_j, otherwise $a_{i,j} = 0$.

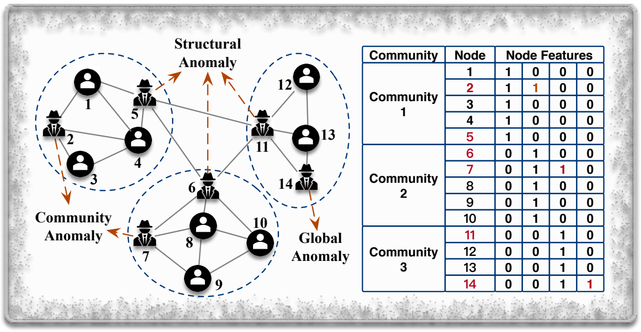
Definition 2 (Attributed Graph). A static attributed graph $G = \{V, E, X\}$ consists of a node set V, an edge set E and an attribute set X. In an attributed graph, the graph structure follows the definition in Definition 1. The attribute matrix $X = [\mathbf{x}_i]_{n\times k}$ consists of nodes' attribute vectors, where $\mathbf{x}_i$ is the attribute vector associated with node v_i and k is the vector's dimension. In the rest of this paper, attribute and feature are used interchangeably.

Definition 3 (Dynamic Graph). A dynamic graph $G(t) = \{V(t), E(t), X_v(t), X_e(t)\}$ consists of nodes and edges changing overtime. $V(t)$ is the set of nodes in the graph at time step t, $E(t)$ is the corresponding edge set, $X_v(t)$ and $X_e(t)$ are the node attribute matrix and edge attribute matrix at time step t in the graph if existed.

In reality, the nodes or edges might also be associated with numerical or categorical labels in these graphs, indicating their classes (e.g., normal or abnormal). When the label information is available/partially-available, supervised/semi-supervised detection models could be effectively trained.

3 Anomalous node detection (ANOS ND)

Anomalous nodes are commonly recognized as individual nodes that are significantly different from others. In real-world applications, these nodes often represent abnormal objects that appear individually, such as a single network intruder in computer networks, an independent fraudulent user in online social networks or a specific fake news in social media. In this section, we particularly focus on anomalous node detection in static graphs, and the reviews on dynamic graphs can be found in Section 4. Table 2 at the end of Section 4 provides a summary of techniques reviewed for ANOS ND.



Community	Node	Node Features			
Community 1	1	1	0	0	0
	2	1	1	0	0
	3	1	0	0	0
	4	1	0	0	0
	5	1	0	0	0
Community 2	6	0	1	0	0
	7	0	1	1	0
	8	0	1	0	0
	9	0	1	0	0
	10	0	1	0	0
Community 3	11	0	0	1	0
	12	0	0	1	0
	13	0	0	1	0
	14	0	0	1	1

Fig. 3: Three Types of Anomalous Nodes: Structural Anomalies, Community Anomalies and Global Anomalies.

When detecting anomalous nodes in static graphs, the differences between anomalies and regular nodes are mainly drawn from the graph structural information and nodes/edges' attributes [41], [71]–[73]. Given the prior knowledge (i.e., community structure, attributes) about a static graph, anomalous nodes can be further categorized into the following three types:

- **Global anomalies** only consider the node attributes. They are nodes that have attributes significantly different from all other nodes in the graph.
- **Structural anomalies** only consider the graph structural information. They are abnormal nodes that have different connection patterns (e.g., connecting different communities, forming dense links with others).
- **Community anomalies** consider both the node attributes and graph structural information. They are defined as nodes that have different attribute values compared to other nodes in the same community.

In Fig. 3, node 14 is a global anomaly because its 4th feature values 1 while all other nodes in the graph have the value of 0 for the corresponding feature. Nodes 5, 6, and 11 are identified as structural anomalies because they have links with other communities while other nodes in their community do not form cross-community links. Nodes 2 and 7 are community anomalies because their feature values are different from others in their belonging communities.

3.1 ANOS ND on Plain Graphs

The plain graphs are dedicated to represent the structural information in real-world networks. To detect anomalous nodes in plain graphs, the graph structure has been extensively exploited from various angles. Here, we first summarize the representative traditional non-deep learning approaches, followed by a more recent, advanced detection technique based on representation learning.

3.1.1 Traditional Non-Deep Learning Techniques

Traditional non-deep learning techniques have been widely used in many real-world networks to identify anomalous individuals before the recent advances in deep learning and other state-of-the-art

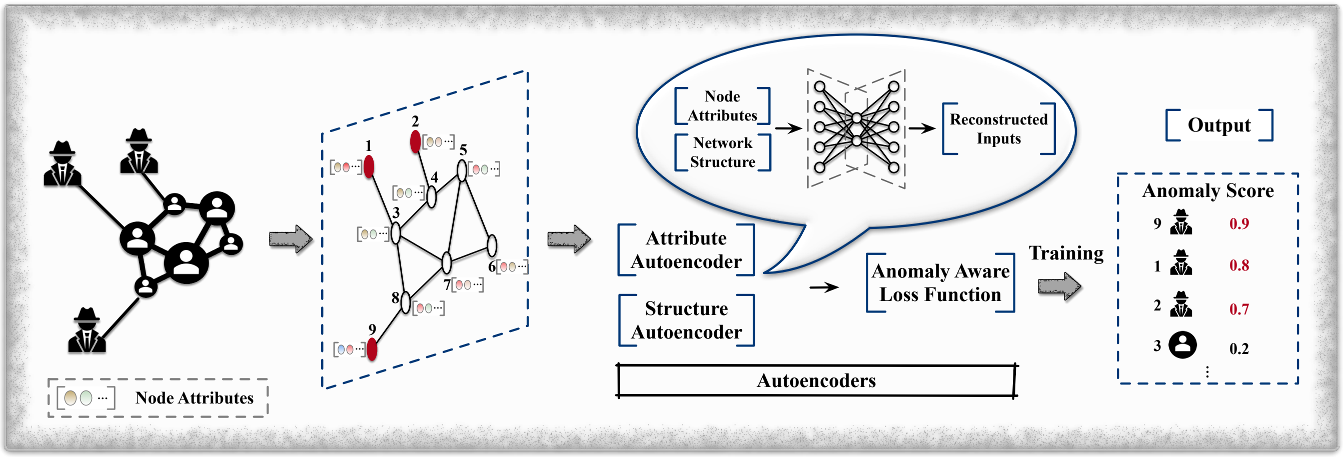


Fig. 4: ANOS ND on attributed graphs – Deep NN based approaches. As an example, the autoencoder is utilized to capture the graph structure and node attributes. With specially-designed anomaly aware loss function, anomaly scores will be assigned to every node and the top-k nodes are anomalies (e.g., nodes 9, 1, and 2 at top-3).

data mining technologies. A key idea behind that is to transform the graph anomaly detection into a traditional anomaly detection problem, because the graph data with rich structure information can not be handled by the traditional detection techniques (for tabular data only) directly. To bridge the gap, lots of works [38], [74], [75] manage to utilize the statistical features associated with each node, such as in/out degree, to detect anomalous nodes.

For instance, OddBall [38] 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 particular 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.

With properly selected statistical features, anomalous nodes can be identified with respect to their deviating feature patterns. But, in real scenarios, it is very hard to choose the most suitable features from a large amount of candidates, and domain experts can always design new statistics, e.g., the maximum/minimum weight of edges. As a result, these techniques introduce prohibitive costs for assessing the most significant features and cannot capture the structural information effectively.

3.1.2 Network Representation Based Techniques

To capture more valuable information from the graph structure for anomaly detection, network representation techniques have been widely exploited. Typically, these techniques encode the graph structure into an embedded vector space and identify anomalous nodes through further analysis. For example, Hu et al. [76] proposed an effective embedding method to detect structural anomalies that are connecting with many communities. It first adopts a graph partitioning algorithm (e.g., METIS [77]) to group nodes into d communities (d is a user-specified number). Then, the method employs a specially designed embedding procedure to learn node embeddings that could capture the link information between each node and d communities. Denoting the embedding for node i as $Z_i = \{z_i^1, \cdots, z_i^d\}$, the procedure initializes each $z_i^c \in Z_i$ with regard to the membership of node i to community c (if node i belongs to community, then $z_i^c = \frac{1}{\sqrt{2}}$; otherwise, 0.) and optimizes node embeddings such that directly linked nodes have similar embeddings and unconnected nodes are dissimilar.

After generating the node embeddings, the link information between node i and d communities can be quantified for further anomaly detection analysis. For a given node i, such information is represented as:

$$\overline{NB(i)} = (y_i^1, ..., y_i^d) = \sum_{j \in NB(i)} (1 - \|Z_i - Z_j\|) \cdot Z_j, \quad (1)$$

where $NB(i)$ is the set of node i's neighbors. If i has many links with community c, then the value in the corresponding dimension y_i^c will be large.

In the last step, [76] formulates a scoring function which can be denoted as:

$$AScore(i) = \sum_{k=1}^{d} \frac{y_i^k}{y_i^*}, y_i^* = \max\{y_i^1, ..., y_i^d\}, \quad (2)$$

to assign anomalousness scores. As expected, the structural anomalies will get higher scores as they connect to different communities. Indeed, given a predefined threshold, nodes which have above-threshold scores are identified as anomalies.

To date, many plain network representation methods such as Deepwalk [78], Node2Vec [79] and LINE [80] have shown their effectiveness in generating node representations and been used for anomaly detection performance validation [81]–[84]. By pairing the conventional anomaly detection techniques such as density-based techniques [85] and distance-based techniques [86] with node embedding techniques, anomalous nodes could be identified with regard to their distinguishable locations (i.e., low-density areas or far away from the majorities) in the embedding space.

3.2 ANOS ND on Attributed Graphs

Apart from the structural information, real-world networks also contain affluent attribute information affiliated with nodes [87], [88]. These attributes provide complementary information about real objects and together with graph structure, more hidden anomalies that are non-trivial can now be detected.

For clarity, we distinguish between deep neural networks and graph neural networks in this survey. We review deep neural network (Deep NN) based techniques, GCN based techniques, and reinforcement learning based techniques for ANOS ND as follows.

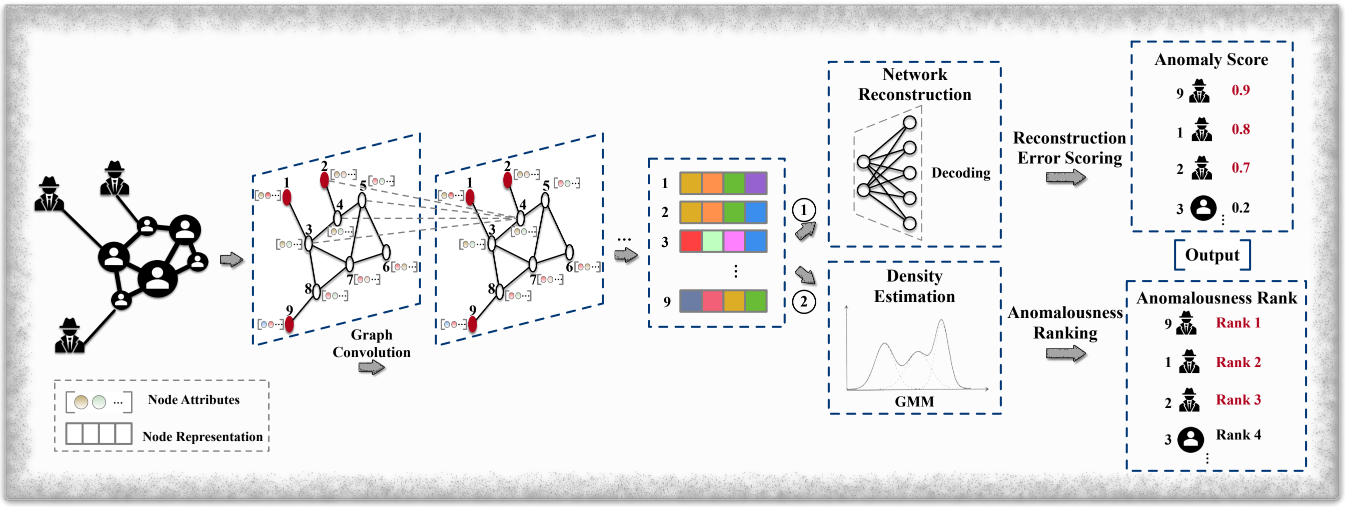


Fig. 5: ANOS ND on attributed graphs – GCN based approaches. Node representations are generated through GCN layers. Anomalies are then detected according to their reconstruction loss (①) or embedding distribution in the embedding space (②).

Due to the page limitation, other existing works including traditional non-deep learning techniques, GAT based techniques, GAN based techniques, and network representation based techniques are surveyed in Appendix C.

3.2.1 Deep NN Based Techniques

The deep learning models such as autoencoder and deep neural networks provide solid basis for learning data representations. Adopting these models for more effective anomalous node detection have drawn substantial interest recently.

For example, Bandyopadhyay et al. [81] developed an unsupervised deep model, DONE, to detect global anomalies, structural anomalies and community anomalies in attributed graphs. Specifically, this work measures three anomaly scores for each node that indicate the likelihood of the situations where 1) it has similar attributes with nodes in different communities (o_i^a), or 2) it connects with other communities (o_i^s), or 3) it belongs to one community structurally but the attribute follow the pattern of another community (o_i^{com}). If a particular node exhibits any of these characteristics, then it is assigned higher score and is anomalous.

To acquire these scores, DONE adopts two separate autoencoders (AE), i.e., structure AE and attribute AE (as shown in Fig. 4), and they are trained by minimizing the reconstruction errors and preserve the homophily that assumes connected nodes to have similar representations in the graph. When training the AEs, nodes exhibiting the predefined characteristics are hard to reconstruct and therefore introduce more reconstruction errors because their structure or attribute patterns do not conform to the prevalent behavior. Hence, the adverse impact of anomalies should be alleviated to achieve the minimized error. Accordingly, DONE specially designs an anomaly aware loss function which has five terms, i.e., $\mathcal{L}_{str}^{Recs}$, $\mathcal{L}_{attr}^{Recs}$, $\mathcal{L}_{str}^{Hom}$, $\mathcal{L}_{attr}^{Hom}$, and $\mathcal{L}^{Com}$. $\mathcal{L}_{str}^{Resc}$, $\mathcal{L}_{attr}^{Resc}$ are the structure reconstruction error and attribute reconstruction error that can be written as:

$$\mathcal{L}_{str}^{Recs} = \frac{1}{N}\sum_{i=1}^{N}\log(\frac{1}{o_i^s})\|\mathbf{t}_i - \hat{\mathbf{t}}_i\|_2^2, \tag{3}$$

and

$$\mathcal{L}_{attr}^{Recs} \frac{1}{N}\sum_{i=1}^{N}\log(\frac{1}{o_i^a})\|\mathbf{x}_i - \hat{\mathbf{x}}_i\|_2^2, \tag{4}$$

where N is the number of nodes, $\mathbf{t}_i$ and $\mathbf{x}_i$ stores the structure information and attributes of node i, $\hat{\mathbf{t}}_i$ and $\hat{\mathbf{x}}_i$ are the reconstructed vectors. $\mathcal{L}_{str}^{Hom}$ and $\mathcal{L}_{attr}^{Hom}$ are proposed to maintain the homophily and they are formulated as:

$$\mathcal{L}_{str}^{Hom} = \frac{1}{N}\sum_{i=1}^{N}\log(\frac{1}{o_i^s})\frac{1}{|N(i)|}\sum_{j\in N(i)}||\mathbf{h}_i^s - \mathbf{h}_j^s||_2^2, \tag{5}$$

and

$$\mathcal{L}_{attr}^{Hom} = \frac{1}{N}\sum_{i=1}^{N}\log(\frac{1}{o_i^a})\frac{1}{|N(i)|}\sum_{j\in N(i)}||\mathbf{h}_i^a - \mathbf{h}_j^a||_2^2, \tag{6}$$

where $\mathbf{h}_i^s$ and $\mathbf{h}_i^a$ are the learned latent representations from the structure AE and attribute AE, respectively. $\mathcal{L}^{Com}$ poses further restrictions on the generated representations for each node by the two AEs such that the graph structure and node attributes could complement each other. It is formulated as:

$$\mathcal{L}^{Com} = \frac{1}{N}\sum_{i=1}^{N}\log(\frac{1}{o_i^{com}})||\mathbf{h}_i^s - \mathbf{h}_i^a||_2^2, \tag{7}$$

By minimizing sum of these loss functions, the anomaly scores of each node are quantified and the top-k nodes with higher scores are identified as anomalies.

3.2.2 GCN Based Techniques

Graph convolutional neural networks (GCNs) [89] 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 GCNs. Fig. 5 illustrates a general framework of existing works in this line.

In [90], Ding et al. measured an anomaly score for each node using the network reconstruction errors of both the structure and attribute. The proposed method, DOMINANT, 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. The structure reconstruction decoder tends to reconstruct the network structure from the learned node embeddings while the attribute reconstruction decoder is proposed to reconstruct the node attribute matrix. The whole neural network is trained to minimize the following loss function:

$$\begin{aligned}\mathcal{L}_{DOMINANT} &= (1-\alpha)\mathcal{R}_S + \alpha\mathcal{R}_A \\ &= (1-\alpha)||A-\hat{A}||_F^2 + \alpha||X-\hat{X}||_F^2,\end{aligned} \tag{8}$$

where α is the coefficient, A depicts the adjacency matrix of the graph, $\mathcal{R}_S$ and $\mathcal{R}_A$ are the structure and attribute reconstruction error, respectively. When the training is finished, an anomaly score is then dedicated to each node according to its contribution to the total reconstruction error and can be calculated using:

$$score(\mathbf{i}) = (1-\alpha)||\mathbf{a}_i - \hat{\mathbf{a}}_i||_2 + \alpha||\mathbf{x}_i - \hat{\mathbf{x}}_i||_2, \tag{9}$$

where $\mathbf{a}_i$ and $\mathbf{x}_i$ are the structure vector and attribute vector of node i, $\hat{\mathbf{a}}_i$ and $\hat{\mathbf{x}}_i$ are their corresponding reconstructed vectors. Thus, by ranking nodes according to their anomaly scores in descending order, the top-k nodes are recognized as anomalies.

To enhance the performance of anomalous node detection, later work by Peng et al. [91] further explores node attributes from multiple attributed views to detect anomalies. The multiple attributed views are employed to describe different perspectives of the objects [92]–[94]. For example, in online social networks user's demographic information and posted contents are two different attributed views and they characterize the personal information and social activities, respectively. The underlying intuition of investigating different views is that anomalies might appear to be normal in one view but abnormal in another view.

For the purpose of capturing these signals, the proposed method, ALARM, applies multiple GCNs 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 [90] aiming at minimizing the network reconstruction loss and attribute reconstruction loss, and can be formulated as:

$$\begin{aligned}\mathcal{L}_{ALARM} =& \sum_{i=1}^{n}\sum_{j=1}^{n} -[\gamma A_{ij}\log\hat{A}_{ij} + (1-A_{ij})\log(1-\hat{A}_{ij})] \\ &+ ||X-\tilde{X}||_2^F,\end{aligned} \tag{10}$$

where γ is coefficient to balance the errors, A_{ij} is the element at coordinate (i,j) in the adjacency matrix A, $\hat{A}_{ij}$ is the corresponding element in the reconstructed adjacency matrix $\hat{A}$, X is the original node feature matrix and $\tilde{X}$ is the reconstructed node feature matrix. Lastly, ALARM adopts the same scoring function as [81], and nodes with top-k highest scores are anomalous.

Instead of spotting unexpected nodes using their reconstruction errors, Li et al. [95] proposed SpecAE 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. The parameters in the GMM is then estimated using the node representations. Due to the deviating attribute patterns of global and community anomalies, normal nodes are expected to exhibit greater energies in GMM and the top-k lowest-probability nodes are anomalies.

In [96], Wang et al. developed a novel detection model that can identify fraudsters using their relations and features. Their proposed method, Fdgars, 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 GCN is trained in a semi-supervised manner by using the user-user network, user features, and labeled users. After training, the model can label unseen users directly.

A more recent work, GraphRfi [97], 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 provide more accurate recommendations to service benign users by alleviating the impact of these untrustworthy users. Specifically, a GCN framework is deployed to encode users and items into a shared embedding space for recommendation and users are classified as fraudsters or normal users through an additional neural random forest using their embeddings. For rating prediction between users and items, the framework reduces the corresponding impact of suspicious users by assigning less weights to their training loss. In the meantime, the rating behavior of users also provide auxiliary information for fraudster detection. The mutual beneficial relationship between these two applications (anomaly detection and recommendation) indicates the potential of information sharing among multiple graph learning tasks.

3.2.3 Reinforcement Learning Based Techniques

The success of reinforcement learning (RL) in tackling real-world decision making problems has gained substantial interests by the anomaly detection community. Detecting anomalous nodes can be naturally regarded as to decide the class (i.e., anomaly or benign) of nodes. Motivated by this, Ding et al. [98] investigated to the use of RL for anomalous node detection. This work also interactively involves human knowledge to verify the anomalies detected by the proposed algorithm, GraphUCB, and achieves enhanced performance. Specifically, GraphUCB models both the attribute information and structural information, and inherits the merits of the contextual multi-armed bandit technology [99] to output potential anomalies. By grouping nodes into K clusters based on their features, GraphUCB forms a K-armed bandit model and measures the payoff of selecting a specific node as a potential anomaly for expert evaluation. With experts' feedback on the predicted anomalies, the decision making strategy is continuously optimized. Eventually, the most potential anomalies can be selected.

A more recent work in [100] intuitively combines reinforcement learning and network embedding techniques to solve the general selective harvesting task where the number of targets is relatively small. As a representative scenario of this task, the anomalous node detection problem could also be solved by the proposed model, NAC. In contrast to GraphUCB, NAC is trained with labeled data without any human intervention. Specifically, it first selects a seed network consisting of partially observed nodes and edges in the whole graph. Then, starting from the seed network, NAC adopts reinforcement learning to learn a node selection plan such that anomalous nodes in the undiscovered area can be identified. This is achieved by rewarding selection plans that can choose labeled anomalies with higher gains. Through offline training, NAC will learn an optimal/suboptimal anomalous node selection strategy and discover potential anomalies in the undiscovered graph step by step.

4 ANOS ND ON DYNAMIC GRAPHS

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, e.g., the evolving patterns of the graph structure and node attributes. On the one hand, these information inherently makes anomalous node detection on dynamic graphs more challenging. This is because dynamic graphs usually introduce large volume of data and temporal signals should also be captured for anomaly detection. On the other hand, they could provide more details about anomalies [25], [47], [112]. For instance, some anomalous nodes might appear to be normal in the graph snapshot at each time stamp, only when the graph structure changes are considered, they can be detected.

In this section, we review the network representation based techniques and GAN based techniques as follows. Other traditional non-deep learning based techniques are reviewed in Appendix D.

4.1 Network Representation Based Techniques

Following the research line of encoding graph into an embedding space, after which anomaly detection is performed, dynamic network representation techniques have been investigated in the more recent works. Specifically, in [83], Yu et al. presented a flexible deep representation technique, NetWalk, to detect anomalous nodes in dynamic (plain) graphs using only the structure information. It adopts an autoencoder 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 [113] 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. When node representations are updated, the cluster centers and anomaly scores are re-calculated accordingly.

4.2 GAN Based Techniques

In practice, anomaly detection is facing great challenges from the shortage of ground-truth anomalies. Consequently, many research efforts have been invested in modeling the features of anomalies or regular objects such that anomalies can be identified effectively.

Among these techniques, generative adversarial networks (GAN) [114] have received extensive attention because of its impressive performance in capturing real data distribution and generating simulated data.

Motivated by the recent advances in “bad” GAN [115], Zheng et al. [111] 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 who behave significantly different. The proposed method, OCAN, 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. A long short-term memory (LSTM) based autoencoder [116] is employed to achieve this and as assumed, benign users and malicious users are in separate regions in the feature space. Next, a novel one-class adversarial net, which contains a generator and a discriminator, is trained. Specifically, the generator targets on generating complementary data points that locate in the relatively low density areas of benign users. The discriminator, accordingly, aims to distinguish these generated samples from the benign users. After training, benign users’ regions are learned by the discriminator and anomalies can hence be identified with regard to their locations.

Both NetWalk [83] and OCAN [111] approach the anomalous node detection problem promisingly, however, they respectively only consider the structure or attributes. By the success of static graph anomaly detection techniques that analyze both aspects, when the structure and attribute information in dynamic graphs are jointly considered, an enhanced detection performance can be foreseen. We therefore highlight this unexplored area for future works in Section 10.

5 ANOMALOUS EDGE DETECTION (ANOS ED)

In contrast to anomalous node detection which targets on individual nodes, ANOS ED aims to identify abnormal links. These links often inform the unexpected or unusual relationships between real objects [117], such as the abnormal interactions between fraudsters and benign users shown in Fig. 1, or suspicious interactions between attacker nodes and benign user machines in computer networks. Following previous taxonomy, we review the state-of-art ANOS ED methods that are built on deep NN, GCN and other network representation techniques for static graphs in this section and dynamic graph techniques are summarized in Section 6. A summary of these techniques is provided in Table 3. For reference, we briefly present traditional non-deep learning based techniques in Appendix E.

5.1 Deep NN Based Techniques

Similar to deep NN based ANOS ND techniques, autoencoder and fully connected network (FCN) have also been applied for anomalous edge detection. As an example, Ouyang et al. [118] approached the problem by modeling the distribution of edges through these deep models and identify existing edges that are least likely to appear as anomalies (as shown in Fig. 6). For each edge (u, v), its probability is decided by $P(v|u, N(u))$ and $P(u|v, N(v))$ which measure the edge probability using node u with its neighbors $N(u)$ and node v with its neighbors $N(v)$, respectively. These two conditional probability measurements are proposed since $N(u)$ and $N(v)$ are different.

To acquire $P(v|u, N(u))$, the proposed method, UGED, first encodes each node into a lower-dimensional vector through a FCN layer and generates node u's representation by a mean aggregation of itself and neighbors’ vectors. Next, the node representations are fed into another FCN to estimate $P(v|u, N(u))$ and the prediction can be denoted as $\hat{P}(v|u, N(u)) = \text{Softmax}(W \cdot H(u))|_v$, where W are trainable parameters, $H(u)$ is u's representation. The training of UGED aims to maximize the prediction of existing edges and a cross-entropy based loss function, $\text{CE}(\hat{P}(v|u, N(u)), v)$, is employed. After training, an anomaly score is assigned to each edge using the average of $1 - P(v|u, N(u))$ and $1 - P(u|v, N(v))$. As such, existing edges that have lower probability will get higher scores and the top-k edges are reported as anomalous.

TABLE 2: Summary of Anomalous Node Detection Techniques.

Graph Type	Approach	Category	Objective Function	Measurement	Outputs
Static Graph - Plain	[76]	NR	$\sum_{(i,j)\in E} \|\mathbf{Z_i}-\mathbf{Z_j}\|^2 + \alpha \sum_{(i,j)\notin E} (\|\mathbf{Z_i}-\mathbf{Z_j}\|-1)^2$	Anomaly Score	$\sum_{k=1}^{d} \frac{y_i^k}{y_i^*}$
	NAC [100]	RL	Cumulative reward	-	Anomalies
Static Graph - Attributed	ALAD [101]	Non-DP	$\min_{W,H} \|A-WW^T\|_F^2 + \alpha\|X-WH\|_F^2 + \gamma(\|W\|_F^2+\|H\|_F^2)$	Anomaly Score	$\frac{\mathbf{W}_{n,c}}{\sum_c \mathbf{W}_{n,c}} cos(\mathbf{A}_{n*}, \mathbf{H}_{c*})$
	Radar [41]	Non-DP	$\min_{W,R} \|X-W^TX-R\|_F^2 + \alpha\|W\|_{2,1} + \beta\|R\|_{2,1} + \gamma tr(R^TLR)$	Residual Analysis	Residual Value
	ANOMALOUS [102]	Non-DP	$\min_{W,\tilde{R}} \|\|X-XWX-\tilde{R}\|\|_F^2 + \alpha\|\|W\|\|_{2,1} + \beta\|\|W^T\|\|_{2,1} + \gamma\|\|\tilde{R}^T\|\|_{2,1} + \varphi tr(\tilde{R}L\tilde{R}^T)$	Residual Analysis	Residual Value
	SGASD [103]	Non-DP	$\sum_{i=0}^{d}\sum_{j=1}^{n_i} \|\|c_{G_j^i}\|\|_2$	Anomaly Prediction	Predicted Label
	DONE [81]	DNN	$\alpha_1\mathcal{L}_{str}^{Recs} + \alpha_2\mathcal{L}_{attr}^{Recs} + \alpha_3\mathcal{L}_{str}^{Hom} + \alpha_4\mathcal{L}_{attr}^{Hom} + \alpha_5\mathcal{L}^{Com}$	Anomaly Scores	o_i^s, o_i^a, o_i^{com}
	DOMINANT [90]	GCN	$(1-\alpha)\mathcal{R}_S + \alpha\mathcal{R}_A$	Anomaly Score	$(1-\alpha)\|\|a_i-\hat{a}_i\|\|_2 + \alpha\|\|x_i-\hat{x}_i\|\|_2$
	ALARM [91]	GCN	$\sum_{i=1}^{n}\sum_{j=1}^{n} -[\gamma A_{ij}\log\hat{A}_{ij} + (1-A_{ij})\log(1-\hat{A}_{ij})]$	Anomaly Score	$(1-\alpha)\|\|a_i-\hat{a}_i\|\|_2 + \alpha\|\|x_i-\hat{x}_i\|\|_2$
	SpecAE [95]	GCN	$\mathbb{E}[dis(X,\hat{X})] + \mathbb{E}[dis(X,\tilde{X})] + \mathbb{E}(E(Z))] + KL$	Density Estimation	Anomalousness Rank
	Fdgars [96]	GCN	$\mathcal{L}_{GCN}$	Anomaly Prediction	Predicted Label
	GraphRfi [97]	GCN	$\frac{1}{\mathcal{U}} \sum_{\forall u\in\mathcal{U}, y_u\in\mathcal{Y}} -\log\mathbb{P}_T[y = y_u\|\mathbf{z}_u^*, \Theta, \pi]$	Anomaly Prediction	Predicted Label
	GraphUCB [98]	RL	Expert Judgment	-	Anomalies
	AnomalyDAE [104]	GAT	$\alpha\|\|(A-\hat{A})\odot\boldsymbol{\theta}\|\|_2^2 + (1-\alpha)\|\|(X-\hat{X})\odot\boldsymbol{\eta}\|\|_2^2$	Reconstruction Loss	Anomalousness Rank
	SemiGNN [105]	GAT	$\alpha\mathcal{L}_{sup} + (1-\alpha)\mathcal{L}_{unsup} + \lambda\mathcal{L}_{reg}$	Anomaly Prediction	Predicted Label
	AEGIS [106]	GAN	$\mathcal{L}_{AE} + \mathcal{L}_{GAN}$	Anomaly Score	$1-D(z_i)$
	REMAD [107]	NR	$\mathcal{L}_{res} + \beta\|R^T\|_{2,1}$	Residual Analysis	Residual Value
	CARE-GNN [27]	NR	$\mathcal{L}_{sim} + \lambda_1\mathcal{L}_{cls} + \lambda_2\mathcal{L}_{reg}$	Anomaly Prediction	Predicted Label
	SEANO [108]	NR	$-\sum_{i\in V_L}\log p(y_i\|x_i,\bar{x}_{N_i}) - \sum_{i\in V}\sum_{v\prime\in C_i}\log p(v\prime\|x_i,\bar{x}_{N_i})$	Anomaly Score	Discriminator's Output
	OCGNN [109]	NR	$\frac{1}{\beta K}\sum_{v_i\in\mathbf{V}_{tr}} [\|\|g(X,A;\mathcal{W})_{v_i} - c\|\|^2 - r^2]^+ + r^2 + \frac{\lambda}{2}\sum_{l=1}^{L}\|\|W^{(l)}\|\|^2$	Location in Embedding Space	Distance to Hypersphere Center
	GAL [66]	NR	$\max\{0, \max_{y_{v'}\neq y_u} g(u,v') - \min_{y_v=y_u} g(u,v) + \Delta_{y_u}\}$	Anomaly Prediction	Predicted Label
Dynamic Graph - Plain	NetWalk [83]	DNN	$\mathcal{L}_{AE} + \mathcal{L}_{Clique} + \|W\|_F^2 + KL$	Anomaly Score	Nearest Distance to Cluster Centers
Dynamic Graph - Attributed	MTHL [110]	Non-DP	$\sum_v\sum_i \tau_i^v\|P^{vT}X_i^vQ^v - Y^*\|_F^2$	Anomaly Score	Distance to Hypersphere Centroid
	OCAN [111]	GAN	$\mathcal{L}_{LSTM} + \mathcal{L}_{Autoencoder}$	Anomaly Score	Discriminator's Output

* Non-DP: Non-Deep Learning, DNN: Deep NN Based techniques, GCN: GCN Based Techniques, RL: Reinforcement Learning.
* GAT: GAT Based Techniques, NR: Network Representation.

5.2 GCN Based Techniques

Following the line of modeling edge distribution for ANOS ED, other works alternatively leverage GCN to better capture the graph structure information. Duan et al. [119] demonstrated that the existence of anomalous edges in the training data prevent traditional GCN based models from capturing the real edge distribution, leading to sub-optimal detection performance. This inherently brings up a problem: to achieve better detection performance, the node embedding process should alleviate the negative impact of anomalous edges, while these edges are detected using the learned embeddings. To tackle this, the proposed method, AANE, jointly considers these two issues by iteratively updating the embeddings and detection results during training.

In each training iteration, AANE generates node embeddings Z through GCN layers and learns an indicator matrix I to spot potential anomalous edges. Given an input graph G with adjacency matrix A, each term I_{uv} in I is 1 if $\hat{A}_{uv} < \text{mean}_{v' \in N_u} \hat{A}_{uv'} - \mu \cdot \text{std}_{v' \in N_u} \hat{A}_{uv'}$, where $\hat{A}_{uv}$ is the predicted link probability between node u and v, which is calculated as the hyperbolic tangent of u and v's embeddings, μ is a predefined threshold,

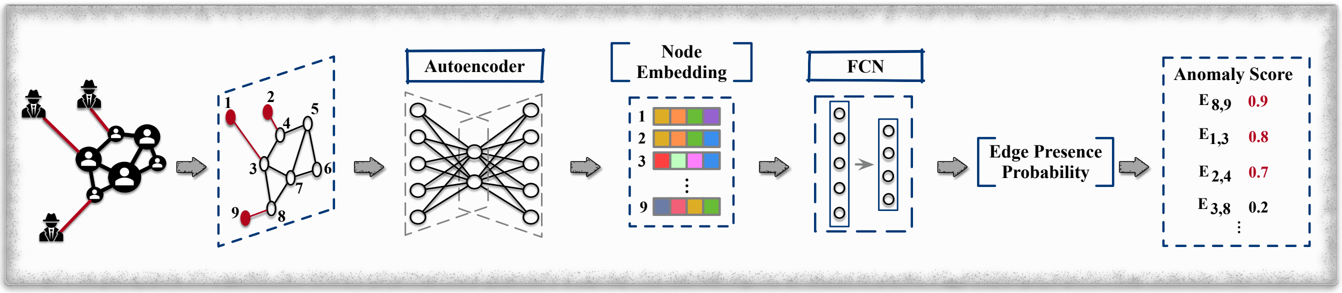


Fig. 6: ANOS ED on static graphs – Deep NN based approaches. For example, the detection technique employs the autoencoder and fully connected network to learn the presence probability of each edge. Anomaly scores are assigned with regard to the probabilities and the top-k edges are anomalous (e.g., $E_{8,9}$, $E_{1,3}$, and $E_{2,4}$ at top-3).

otherwise, 0. By this, 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.

The total loss function of AANE contains two parts: an anomaly aware loss ($\mathcal{L}_{aal}$) and an adjusted fitting loss ($\mathcal{L}_{afl}$). $\mathcal{L}_{aal}$ is proposed to penalize the link prediction results and indicator matrix I such that anomalous edges will have lower prediction probabilities where they are marked as 1 in I. This is formulated as:

$$\mathcal{L}_{aal} = \sqrt{\sum_{u \in V} \sum_{v \in N(u)} ((1 - \hat{A}^2_{uv})(1 - I_{uv}) + \hat{A}^2_{uv} I_{uv})}, \quad (11)$$

where V is the node set, $N(u)$ is the set of neighbors of u. $\mathcal{L}_{afl}$ quantifies the reconstruction loss with regard to the removal of potential anomalous edges and it can be denoted as:

$$\mathcal{L}_{afl} = \|B - \hat{A}\|_2^2, \quad (12)$$

where B is an adjusted adjacency matrix that removes all predicted anomalies from the input adjacency matrix A. By minimizing these two losses, AANE will spot the top-k edges with lowest probabilities as anomalies.

5.3 Network Representation Based Techniques

Instead of using node embeddings for ANOS ED, edge representations that are learned directly from the graph is also feasible to apply. If the edge representations could well-preserve the graph structure and interaction content (e.g., messages in online social networks, co-authored papers in citation networks) between pairs of nodes, an enhanced detection performance can then be expected. To date, several studies, such as Xu et al. [120], have shown promising results in generating edge representations. Although they are not particularly designed for graph anomaly detection, they pinpoint a potential approach to ANOS ED and we highlight this as a potential future direction in Section 10.1.

6 ANOS ED on Dynamic Graphs

Dynamic graphs are powerful in reflecting the appearance/disappearance of edges over time [121]. By modeling the changes in graph structure and capturing the regular edge distribution at each time step, anomalous edges can be distinguished with regard to their unusual occurrences. Recently, several works have leveraged deep learning techniques to approach ANOS ED and we review them in this section.

6.1 Network Representation Based Techniques

The intuition of network representation based techniques is to encode the dynamic graph structure information into edge representations and apply the aforementioned traditional anomaly detection techniques to spot irregular edges. This is quite straightforward, but there remains vital challenges in generating/updating informative edge representations when the graph structure evolves. To mitigate this challenge, the ANOS ND model NetWalk [83] 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 identify anomalies based on their distances to the nearest edge-cluster centers in the latent space. Practically, Netwalk generates edge representations as the Hadamard product of the source and destination nodes' representations, which can be denoted as: $\mathbf{z}_{u,v} = \mathbf{z}_u \odot \mathbf{z}_v$. 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. Finally, the top-k farthest edges to edge-clusters are reported as anomalies.

6.2 GCN Based Techniques

Although NetWalk is capable of detecting anomalies in dynamic graphs, it simply updates edge representations without considering the long/short-term node and graph structure evolving patterns. For more effective ANOS ED, Zheng et al. [122] intuitively combined temporal, structural and attribute information to measure the anomalousness of edges in dynamic graphs. They propose a semi-supervised model, AddGraph, which comprises a GCN and Gated Recurrent Units (GRU) with attention [123] to capture more representative structural information from the temporal graph in each time stamp and dependencies between them, respectively.

At each time stamp t, GCN takes the output hidden state (H^{t-1}) at time $t-1$ to generate node embeddings, after which GRU learns the current hidden state H^t from the node embeddings and attention on previous hidden states (as shown in Fig. 7). After getting the hidden state H^t of all nodes, AddGraph assigns an anomaly score to each edge in the temporal graph based on the nodes associated with it. The proposed anomaly scoring function is formed as:

$$f(u, v, w) = w \cdot \sigma(\beta \cdot (||\mathbf{a} \odot \mathbf{h}_u + \mathbf{b} \odot \mathbf{h}_v||_2^2 - \mu)), \quad (13)$$

where u and v are the corresponding nodes, w is the weight of the edge, $\mathbf{a}$ and $\mathbf{b}$ are trainable parameters, β and μ are hyperparameters, and $\sigma(\cdot)$ is the sigmoid function. To learn $\mathbf{a}$ and $\mathbf{b}$,

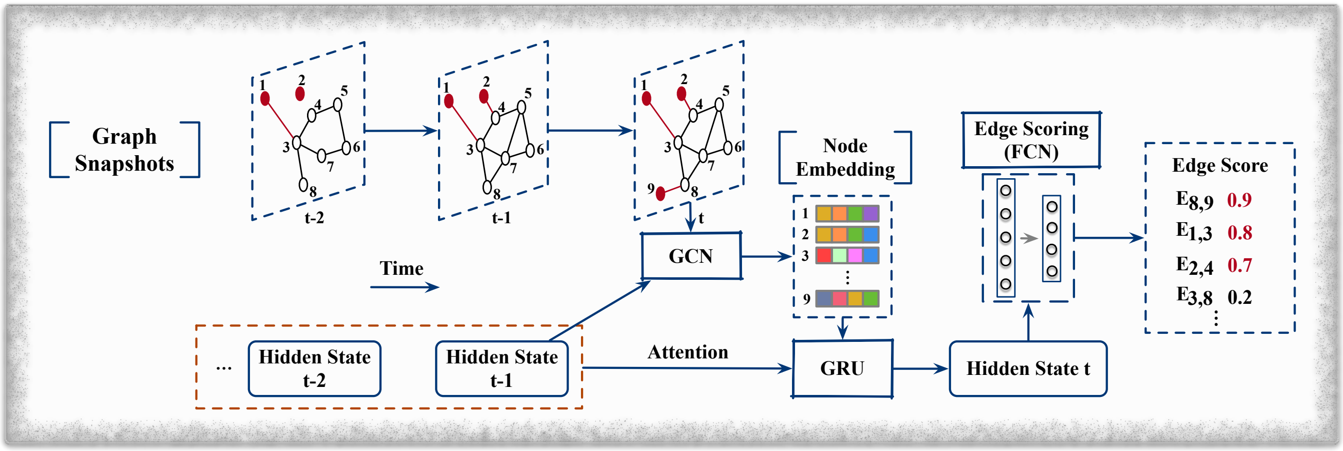


Fig. 7: ANOS ED on dynamic graphs – GCN based approaches. GCN is employed to learn node embeddings from the temporal graph at each time stamp. The attention based GRU generates the current hidden state using the node embeddings and previous hidden states. The edge scoring function, such as a FCN, is learned to assign anomaly scores and the top-k edges are depicted as anomalies.

Zheng et al. further assume that all existing edges in the dynamic graph are normal in the training stage and sample non-existing edges as anomalies. Specifically, they form the loss function as:

$$\mathcal{L}_{AddGraph} = \min \sum_{(u,v,w)\in\varepsilon^t} \sum_{(u',v',w)\notin\varepsilon^t} \max\{0, \gamma + f(u,v,w) - f(u',v',w)\} + \lambda\mathcal{L}_{reg}, \tag{14}$$

where ε^t is the edge set, (u', v') are sampled non-existing edges at time stamp t, λ is a hyper-parameter, and $\mathcal{L}_{reg}$ regularizes all trainable parameters in the model. After training, the scoring function identifies anomalous edges in the test data by assigning higher anomaly scores to them based on Eq. 13.

7 Anomalous sub-graph detection (ANOS SGD)

In real life, anomalies might also collude and behave collectively with others to get more profits. For instance, fraudulent user groups in online review networks (as shown in Fig. 1) facilitate to post misleading reviews to promote or demote certain merchandise or business. When the data are represented by graphs, these anomalies and their interactions usually form suspicious sub-graphs, and ANOS SGD is proposed to distinguish them from the benign.

Different from individual and independent graph anomalies (i.e., single nodes or edges), each node and edge in suspicious sub-graphs might be normal; however, they are anomalous when they are considered together as a collection. Moreover, these sub-graphs also vary in size and inner structure, making anomalous sub-graph detection more challenging than ANOS ND/ED [124]. Although extensive efforts have been put on circumventing this problem, deep-learning techniques have been incorporated only in the last five years. For reference, we briefly introduce traditional non-deep learning based techniques in Appendix F and provide a summary of techniques reviewed for ANOS SGD in Table 3 at the end of Section 8.

Due to the flexibility of heterogeneous graphs that contain more than one type of nodes or edges in representing the complex relationships between different kinds of real objects, several recent works take advantage of and leverage deep network representation techniques to detect real-world anomalies through ANOS SGD. For instance, Wang et al. [125] represent online shopping networks as bipartite graphs (a specific type of heterogeneous graph that has two types of nodes and one type of edge), in which users are source nodes and items are sink nodes, and detect fraudulent groups based on the suspicious dense blocks they have formed in these graphs.

[125] aims to learn anomaly-aware representations of users such that suspicious users in the same group will locate closely in the vector space, while benign users are far away from them (as shown in the embedding space in Fig. 8). According to the observation that user nodes belonging to one fraudulent group are more likely to connect with same item nodes, the developed model, DeepFD, especially measures two users' behavior similarity, sim_{ij}, as the percentage of shared items in all the items they have reviewed. The user representations are then generated through a traditional autoencoder which is trained following the encoding-decoding process and using three losses. The first is the reconstruction loss $\mathcal{L}_{res}$ that ensures the bipartite graph structure can be well-reconstructed using the learned user representations and item representations. The second term $\mathcal{L}_{sim}$ is to preserve the user similarity information in the learned user representations (i.e., if two users have similar behaviors, their representations should also be similar) and this can be denoted as:

$$\mathcal{L}_{sim} = \sum_{i,j=1}^{m} sim_{ij} \cdot \|\widehat{sim}_{ij} - sim_{ij}\|_2^2, \tag{15}$$

where m is the number of user nodes, $\widehat{sim}_{ij}$ measures the similarity of user i and j's representations using RBF kernel or other alternatives. The third loss $\mathcal{L}_{reg}$ is proposed to regularize all trainable parameters. Finally, the suspicious dense blocks which are expected to form dense regions in the vector space and they are detected by DeepFD using DBSCAN [126].

Another work, FraudNE [127], also models online review networks as bipartite graphs and further detects both malicious users and associated manipulated items following the dense block detection principle. Unlike DeepFD, FraudNE aspires to encode both types of nodes into a shared latent space where suspicious users and items belonging to the same dense block are very close to each other while others distribute uniformly (as

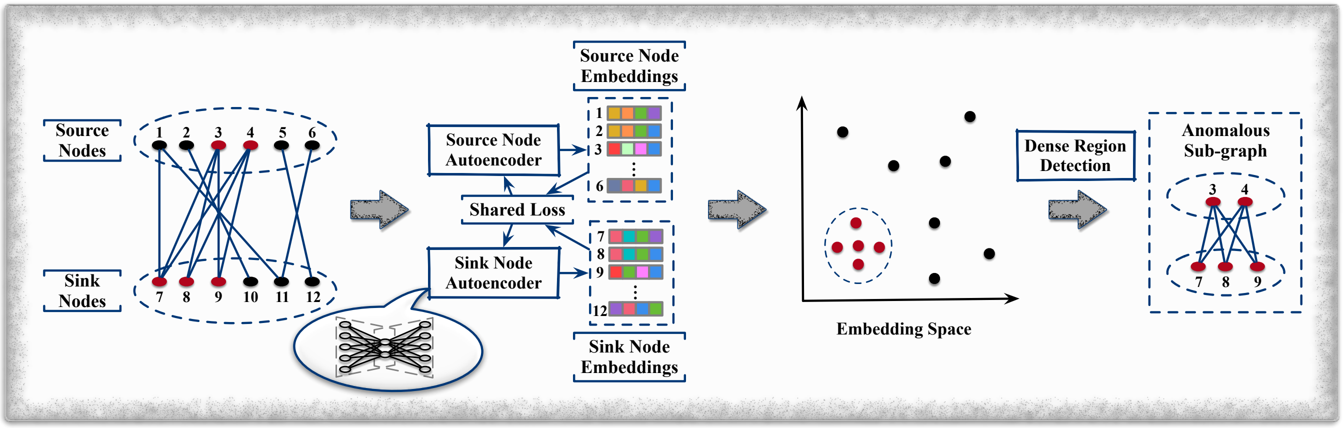


Fig. 8: ANOS SGD. Real world networks are usually represented as bipartite graphs for reflecting the interactions between two different types of objects. To detect ANOS SGD, source nodes and sink nodes are embedded using two autoencoders (linked by a shared loss function), respectively. Anomalous sub-graphs are identified by applying dense region detection algorithms in the embedding space.

shown in Fig. 8). It adopts two traditional autoencoders, namely, source node autoencoder and sink node autoencoder to learn user representations and item representations, respectively. Both autoencoders are trained to jointly minimize their corresponding reconstruction losses and a shared loss function, and the total loss can be denoted as:

$$\mathcal{L}_{FraudNE} = \mathcal{L}_{res}^{source} + \mathcal{L}_{res}^{sink} + \alpha\mathcal{L}_{share} + \eta\mathcal{L}_{reg}, \quad (16)$$

where α and η are hyperparameters, and $\mathcal{L}_{reg}$ regularizes all trainable parameters. Specifically, the reconstruction losses (i.e., $\mathcal{L}_{res}^{source}$ and $\mathcal{L}_{res}^{sink}$) measure the gap between the input user/item features (extracted from the graph structure) and their decoded features. The shared loss function is proposed to restrict the representation learning process such that each linked pair of user and item get similar representations. As the DBSCAN [126] algorithm is convenient to apply for dense region detection, FraudNE also adopts it to distinguish the dense sub-graphs formed by suspicious users and items.

To date, only a few works have put their efforts into utilizing deep learning techniques for ANOS SGD. However, with the intensified research interests in sub-graph representation learning, we encourage more studies on ANOS SGD and highlight this as a potential future in Section 10.1.

8 Anomalous graph detection (ANOS GD)

Apart from anomalous node/edge/sub-graph, graph anomalies might also appear as abnormal graphs in a sequence/set/database of graphs. Typically, the graph database is defined as:

Definition 4 (Graph Database). A graph database $\mathcal{G} = \{G_i = (V_i, E_i, X_v(i), X_e(i))\}_{i=1}^{N}$ contains N individual graphs. Here, each graph G_i is comprised of a node set V_i and an edge set E_i. $X_v(i)$ and $X_e(i)$ are the node attribute matrix and edge attribute matrix $X_e(i)$ of G_i if it is an attributed graph.

ANOS GD aims to detect these individual graphs that deviate significantly from others. A concrete example of ANOS GD is unusual molecule detection. When chemical compounds are represented as molecular/chemical graphs where the atoms and bonds are represented as nodes and edges [128], [129], unusual molecules can be effectively identified because their corresponding graphs have deviating structures and features to others. Another example is brain disorders detection. Given the brain graphs at different age stages, by analyzing the dynamics (e.g., interaction patterns between brain regions) of the brain graph sequence, a brain disorder can be diagnosed based on the inconsistent brain graph snapshot at a specific time stamp, in which some rarely interacting brain regions are connected.

The prior reviewed graph anomaly detection techniques (i.e., ANOS ND/ED/SGD) are incapable to ANOS GD because they are dedicated to detect anomalies in a single graph while ANOS GD is aiming at graph-level anomalies. Although plenty of works have tried to approach the problem by measuring the pairwise proximities of graphs using graph kernels [130], or by detecting the appearance of anomalous graph signals created by abnormal groups of nodes [131], or via encoding graphs using frequent motifs [62], none of them are deep learning based. Till the time of our survey, very few studies have been performed in ANOS GD with deep learning and we pinpoint it as a potential future direction in Section 10.1.

8.1 Deep NN Based Techniques

Similar to ANOS ND/ED in dynamic graphs modeled as graph sequences, one of the key ideas behind the deep learning-based techniques is to encode the comprehensive information (e.g., graph evolving patterns, features of graph snapshots) contained in the sequence for ANOS GD.

Among commonly used DNNs, LSTM and autoencoder are feasible to digest each graph snapshot's characteristics from the graph snapshot sequences. Specifically, Teng et al. [132] applied both neural networks to detect abnormal graph snapshots that is different from others. In the proposed model, DeepSphere, the dynamic graph is described as a collection of three-order tensors, $\{\mathcal{X}_k, k = 1, 2...\}$ where each $\mathcal{X} \in \mathcal{R}^{N \times N \times T}$, and the slices along the time dimension are the adjacency matrices of graph snapshots. To identify the abnormal tensors, DeepSphere first embeds each graph snapshot into a latent space using the LSTM autoencoder, and then leverages a one-class classification objective [133] that learns a hypersphere such that normal snapshots are covered and anomalous snapshots are lying outside. The LSTM autoencoder takes the adjacency matrices as input sequentially and is trained to reconstruct the input sequence. And the hypersphere is learned through a single neural network layer and its objective

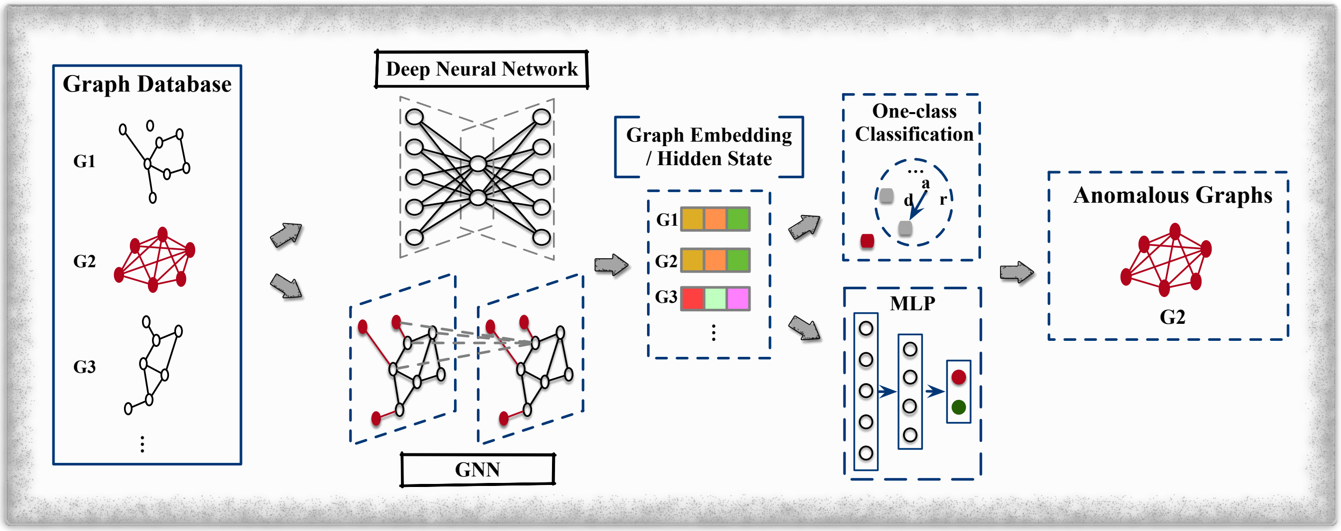


Fig. 9: ANOS GD on Graph Database. Generally, the graph-level anomaly detection techniques take a graph database as input. By generating embeddings/hidden state for each single graph in the database through D(G)NNs, anomalous graphs can be depicted by One-class classifiers or MLP.

function can be denotes as:

$$\mathcal{L}_h = r^2 + \gamma \sum_{k=1}^{m} \epsilon_k + \frac{1}{m} \sum_{k=1}^{m} \|\mathbf{z}_k - \mathbf{a}\|^2, \tag{17}$$

where $\mathbf{z}_k$ is the latent representation generated by the LSTM autoencoder, $\mathbf{a}$ is the centroid of the hypersphere, r is the radius, ϵ_k is the outlier penalty ($\epsilon_k = \|\mathbf{z}_k - \mathbf{a}\|^2 - r^2$), m is the number of training graph snapshots, and γ is a hyperparameter. The overall objective function of DeepSphere can be represented as:

$$\mathcal{L} = \mathcal{L}_h + \lambda \mathcal{L}_{res}, \tag{18}$$

where $\mathcal{L}_{res}$ is the reconstruction loss of the LSTM autoencoder. When the training is finished, for a given unseen data $\mathcal{X}$, DeepSphere spots it as anomalous if its embedding lies outside the learned hypersphere with radius r.

8.2 GNN Based Techniques

ANOS GD also aims to identify anomalous graphs given an unordered set of graphs. Specifically, Dou et al. [134] transformed fake news detection as an ANOS GD problem by modeling news as tree-structured propagation graphs where the root nodes represent the news piece, and other nodes represent users sharing the root news. In the proposed end-to-end framework, UPFD, the users' historical posts, news content as well as the propagation graphs are utilized to fuse user engagement information (user engagement embedding) through GNN layers and to extract news textual embeddings by text representation learning models (e.g. word2vec, BERT). For each news, its latent representation is a flattened concatenation of these two embeddings, which is input to train a neural classifier with the news' label. The corresponding propagation graph that is labeled as fake by the trained model can be regarded as anomalous.

Another representative work by Zhao and Akoglu [135] particularly employed the GIN model and one-class classification (i.e., DeepSVDD [133]) loss to train an end-to-end framework for graph-level anomaly detection. For each individual graph in the graph database, its graph-level embedding is generated by applying mean-pooling over its nodes' node-level embeddings. A graph is eventually depicted as anomalous if it lies outside the learned hypersphere (as shown in Fig. 9).

8.3 Network Representation Based Techniques

The general graph-level network representation techniques are also feasible to apply for ANOS GD by transferring the graph anomaly detection problem as an conventional outlier detection problem in the embedding space. In contrast to D(G)NN based techniques that can detect graph anomalies in an end-to-end manner as aforementioned, adopting the general graph-level representation techniques for anomaly detection is two-staged. Firstly, graphs in the database are encoded into a shared latent space using graph-level representation techniques, such as Graph2Vec [136], FGSD [137]. Then, the anomalousness of each single graph is measured by off-the-shelf outlier detectors. Basically, this kind of approach involves pairing existing methods in both stages, yet, the stages are disconnected from each other and hence the detection performance can be subpar since embeddings similarities are not necessarily designed for the sake of anomaly detection.

In addition to all ANOS ND, ED, SGD, and GD techniques reviewed prior, it is worth mentioning that perturbed graphs, which adversarial models generate to attack graph classification algorithms or GNNs [138]–[140], can also be regarded as (intensional) anomalies. In the perturbed graphs, the nodes and edges are modified deliberately and are deviating from others. Although the adversarial attack models generate these graphs, we did not review them in our survey because they mainly aim to attack the GNN models. The key ideas behind these works are the attacking/perturbation strategies. They seldomly focus on investigating a detection or reasoning module to identify the perturbed graph or its sub-structures, i.e., anomalous nodes, edges, sub-graphs, or graphs.

9 Published Algorithms and Datasets

Acquiring open-sourced implementations and real-world datasets with real-world anomalies are far from trivial in academic research in graph anomaly detection. Here, we first list published

TABLE 3: Summary of Anomalous Edge, Sub-graph and Graph Detection Techniques.

Graph Type	Approach	Category	Objective Function	Measurement	Outputs
Anomalous Edge Detection Techniques					
Static Graph - Plain	UGED [118]	DNN	cross-entropy$(f(u, N(u)), v)$	Anomaly Score	mean$(1 - P(v\|u, N(u)), 1 - P(u\|v, N(v)))$
	AANE [119]	GCN	$\mathcal{L} = \mathcal{L}_{afl} + \gamma\mathcal{L}_{aal}$	Anomaly Ranking	Edge Existing Probability
Dynamic Graph - Plain	NetWalk [83]	NR	$\mathcal{L}_{AE} + \mathcal{L}_{Clique} + \|W\|_F^2 + KL$	Anomaly Score	Nearest Distance to Cluster Centers
Dynamic Graph - Attributed	AddGraph [122]	GCN	min $\sum_{e \in \varepsilon^t} \sum_{e' \notin \varepsilon^t}$ max$\{0, \gamma + f(i,j,w) - f(i',j',w)\} + \lambda\mathcal{L}_{reg}$	Anomaly Score	$f(i,j,w) = w \cdot \sigma(\beta \cdot (\|\|\mathbf{a} \odot \mathbf{h}_i + \mathbf{b} \odot \mathbf{h}_j\|\|_2^2 - \mu))$
Anomalous Sub-graph Detection Techniques					
Static Graph - Plain	DeepFD [125]	NR	$\mathcal{L}_{recon} + \alpha\mathcal{L}_{sim} + \gamma\mathcal{L}_{reg}$	Density-based Method (DBSCAN)	Dense sub-graphs
	FraudNE [127]	NR	$\mathcal{L}_{res}^{source} + \mathcal{L}_{res}^{sink} + \alpha\mathcal{L}_{share} + \eta\mathcal{L}_{reg}$	Density-based Method (DBSCAN)	Dense sub-graphs
Anomalous Graph Detection Techniques					
Graph Database - Plain	UPFD [134]	NR	$-(y\log(p) + (1-y)\log(1-p))$	Anomaly Prediction	Predicted Label
Graph Database - Attributed	OCGIN [135]	GIN	$\min_{W} \frac{1}{N}\sum_{i=1}^{N} \|\|GIN(G_i, W) - c\|\|^2 + \frac{\lambda}{2}\sum_{l=1}^{L} \|\|W^l\|\|_F^2$	Location in Embedding Space	Distance to Hypersphere Center
Dynamic Graph - Plain	DeepSphere [132]	DNN	$\mathcal{L} = \mathcal{L}_h + \lambda\mathcal{L}_{res}$	Location in Embedding Space	Anomalous Label

* DNN: Deep NN Based techniques, GCN: GCN Based Techniques, NR: Network Representation.

algorithms with publicly available implementations related to graph anomaly detection. Then, we provide a collection of public benchmark datasets and summarize the commonly used evaluation metrics. Lastly, due to the shortage of labeled anomalies in real-world datasets, we review three synthetic dataset generation strategies that are adopted in the existing work.

9.1 Published Algorithms

The published implementations of algorithms and models contribute to baseline experiments. We provide a summary of published implementations in Table 4 concerning their implementing language and platforms, graphs that they can admit, and URL to code repository.

9.2 Published Datasets

We summarize the mostly used datasets and categorize them into different groups with regard to the application fields. Details of these datasets are given in Table 5. Specifically, for the lack of ground-truth anomalies, labeled anomalies are provided only in Enron, Twitter Sybil, Disney, Amazon, Elliptic and Yelp datasets. The details of DBLP, UCI message, Digg, Wikipedia, and New York city taxi datasets are not given because these public datasets only contain the raw data, and in most existing works, they are further processed to build different graphs (e.g., homogeneous graphs, bipartite graphs). The well-known citation networks are utilized to generate synthetic datasets by injecting anomalies into them. As existing works inject distinctive number anomalies into these datasets, the number of anomalies varies.

Apart from these anomaly detection datasets, we also collect eight mostly used graph classification datasets in Table 5. Through further processing (i.e., downsampling) mentioned in 9.3, these datasets can be utilized as benchmarks for evaluating the anomaly detection performance.

9.3 Synthetic Dataset Generation

In face of the rarity of ground-truth anomalies, plenty of works have employed synthetic datasets to investigate the effectiveness of their proposed methods [82], [160], [161]. Typically, these datasets can be categorized as follows:

- *Synthetic graph with injected anomalies.* Pursuing this strategy, graphs are created to simulate real-world networks. All the nodes and edges are manually added with well-known benchmarks (e.g., Lanchinetti-Fornunato-Radicchi (LFR) [162], small-world [163], scale-free graphs [164]), after which ground-truth anomalies are planted into the network. For the feasibility of generating expected scale of networks, this strategy is mostly used by previous works to validate their underlying intuitions in anomaly detection.
- *Real-world dataset with injected anomalies.* These datasets are built based on the real-world networks. In particular, anomalies are generated either by perturbing the topological structure or attributes of existing nodes/edges/sub-graphs, or by inserting non-existing graph objects.

TABLE 4: Published Algorithms and Models

Model	Language	Platform	Graph	Code Repository
Sedanspot [141]	C++	-	Dynamic Graph	https://www.github.com/dhivyaeswaran/sedanspot
AnomalyDAE [104]	Python	Tensorflow	Dynamic Attribute Graph	https://github.com/haoyfan/AnomalyDAE
MADAN [142]	Python	-	Static Attributed Graph	https://github.com/leoguti85/MADAN
PAICAN [71]	Python	Tensorflow	Static Attributed Graph	http://www.kdd.in.tum.de/PAICAN/
Changedar [131]	Matlab	-	Dynamic Attributed Graph	https://bhooi.github.io/changedar/
ONE [82]	Python	-	Static Plain Graph	https://github.com/sambaranban/ONE
DONE&AdONE [81]	Python	Tensorflow	Static Attributed Graph	https://bit.ly/35A2xHs
SLICENDICE [143]	Python	-	Static Attributed Graph	http://github.com/hamedn/SliceNDice/
SemiGNN [105]	Python	Tensorflow	Static Attributed Graph	https://github.com/safe-graph/DGFraud
CARE-GNN [27]	Python	Pytorch	Static Attributed Graph	https://github.com/YingtongDou/CARE-GNN
GraphConsis [144]	Python	Tensorflow	Static Attributed Graph	https://github.com/safe-graph/DGFraud
GLOD [135]	Python	Pytorch	Static Attributed Graph	https://github.com/LingxiaoShawn/GLOD-Issues
GCAN [145]	Python	Keras	Heterogeneous Graph	https://github.com/l852888/GCAN
HGATRD [146]	Python	Pytorch	Heterogeneous Graph	https://github.com/201518018629031/HGATRD
GLAN [147]	Python	Pytorch	Heterogeneous Graph	https://github.com/chunyuanY/RumorDetection
ANOMRANK [148]	C++	-	Dynamic Graph	https://github.com/minjiyoon/anomrank
DAGMM [149]	Python	Pytorch	Dynamic Graph	https://github.com/danieltan07/dagmm
OCAN [111]	Python	Tensorflow	Static Graph	https://github.com/PanpanZheng/OCAN
DevNet [150]	Python	Tensorflow	Static Graph	https://github.com/GuansongPang/deviation-network
RDA [151]	Python	Tensorflow	Static Graph	https://github.com/zc8340311/RobustAutoencoder
GAD [152]	Python	Tensorflow	Static Graph	https://github.com/raghavchalapathy/gad
GEM [153]	Python	-	Static Graph	https://github.com/safe-graph/DGFraud/tree/master/algorithms/GEM
MIDAS [154]	C++	-	Dynamic Graph	https://github.com/Stream-AD/MIDAS
DeFrauder [155]	Python	-	Static Graph	https://github.com/LCS2-IIITD/DeFrauder
DeepFD [125]	Python	Pytorch	Bipartite Graph	https://github.com/JiaWu-Repository/DeepFD-pyTorch
STS-NN [156]	Python	Pytorch	Static Graph	https://github.com/JiaWu-Repository/STS-NN
UPFD [134]	Python	Pytorch	Graph Database	https://github.com/safe-graph/GNN-FakeNews
DeepSphere [132]	Python	Tensorflow	Dynamic Graph	https://github.com/picsolab/DeepSphere
OCGIN [135]	Python	Pytorch	Graph Database	https://github.com/LingxiaoShawn/GLOD-Issues
Deep SAD [157]	Python	Pytorch	Non Graph	https://github.com/lukasruff/Deep-SAD-PyTorch
DATE [158]	Python	Pytorch	Non Graph	https://github.com/Roytsai27/Dual-Attentive-Tree-aware-Embedding

* -: No Dedicated Platforms.

- *Downsampling graph classification datasets.* The widely-used graph classification datasets (e.g., NCI1, IMDB, ENZYMES in [135]) can be easily transferred for anomaly detection through two steps. Firstly, one particular class and its data records are chosen to represent normal objects. Then, other data records are down-sampled (with a downsampling rate) as anomalies. By this, the generated GAD dataset is, in fact, a subset of the original dataset, and its most significant pro is that no single data record has been modified.

9.4 Evaluation Metrics

To date, the widely used metrics for anomaly detection performance evaluation include accuracy, precision, recall rate, F1-score and AUC-AP (Average Precision), and their formulas/descriptions are given in Table 6. These metrics are feasible to apply, but more dedicated analysis are needed for further performance examination because anomaly detection in different application fields have divergent requirements on the statistics [165]–[167], e.g., false negative errors and false positive errors. For instance, network intrusion prevention systems is more sensitive to false negative errors while false positive errors are considered relatively less harmful, because any risky connections should be shut down in case of information leakage. In contrast, other applications concentrate more on the false positives, e.g., in auditing domain, companies often set budget for an auditor to look at flagged anomalies, and they want high precision/small false positive rate such that the auditor's time can be best used. Hence, when evaluating the detection performance, we suggest more attention to be put on the application domain for fair and suitable comparisons.

10 Future Directions

So far, we have reviewed the contemporary deep learning techniques that are devoted to graph anomaly detection. An apparent observation from our survey is there remain many compounded challenges imposed by the complexity of anomaly detection, graph data, and immaturity of deep learning techniques for graph data mining. Another observation is the adoption of existing deep learning techniques is still confined to relatively small amount of studies in graph anomaly detection. Most of them are focusing on anomalous node detection (compare the length of Table 2 and Table 3) while edge, sub-graph, and graph-level anomaly detection have received much less attention. To bridge the gaps and push forward future work, we identify and highlight twelve directions for future research on graph anomaly detection with deep learning.

10.1 Anomalous Edge, Sub-graph, and Graph Detection

In real-world graphs, anomalies would also appear as unusual relationships between objects, sub-structures formed by abnormal groups, or abnormal graphs, which are known as anomalous edges, sub-graphs, and graphs respectively. Moreover, in some domains, data may be represented as a set/collection/database of graphs, where the anomaly detection task is to identify abnormal subset of graphs (i.e. graph-level detection). As indicated from our review, there is a huge gap between the existing anomalous edge/sub-graph/graph detection techniques and the emerging demands for more advanced solutions in various application domains (e.g., social networks, computer networks, financial networks). When detecting anomalous edges/sub-graphs/graphs, the proposed methods should be capable of utilizing affluent information contained in graphs to find clues and characteristics that can distinguish

TABLE 5: Published Datasets.

Category	Dataset	#G	#N	#E	#FT	#AN	REF	URL
Citation Networks	ACM	1	16K	71K	8.3K	-	[90], [98], [104], [106]	http://www.arnetminer.org/open-academic-graph
	Cora	1	2.7K	5.2K	1.4K	-	[71], [81], [82], [95], [108]	http://linqs.cs.umd.edu/projects/projects/lbc
	Citeseer	1	3.3K	4.7K	3.7K	-	[12], [81], [82], [108]	http://linqs.cs.umd.edu/projects/projects/lbc
	Pubmed	1	19K	44K	500	-	[81], [82], [95], [108]	http://linqs.cs.umd.edu/projects/projects/lbc
	DBLP	1	-	-	-	-	[12], [71], [76], [83], [141]	http://www.informatik.uni-trier.de/˜ley/db/
Social Networks	Enron	-	80K	-	-	-	[39], [41], [102], [107], [112], [141], [142], [148], [159]	http://odds.cs.stonybrook.edu/#table2
	UCI Message	1	5K	-	-	-	[83], [84], [122]	http://archive.ics.uci.edu/ml
	Google+	4	75M	11G	-	-	-	https://wangbinghui.net/dataset.html
	Twitter Sybil	3	41M	-	-	100K	-	https://wangbinghui.net/dataset.html
	Twitter World-Cup2014	-	54K	-	-	-	[159]	http://shebuti.com/SelectiveAnomalyEnsemble/
	Twitter Security2014	-	130K	-	-	-	[159]	http://shebuti.com/SelectiveAnomalyEnsemble/
	Reality Mining	-	9.1K	-	-	-	[159]	http://shebuti.com/SelectiveAnomalyEnsemble/
	NYTNews	-	320K	-	-	-	[159]	http://shebuti.com/SelectiveAnomalyEnsemble/
	Politifact	314	41K	40K	-	157	[134]	https://github.com/safe-graph/GNN-FakeNews
	Gossipcop	5.4K	314K	308K	-	2.7K	[134]	https://github.com/safe-graph/GNN-FakeNews
Co-purchasing Networks	Disney	1	124	334	30	6	[41], [101], [102], [107], [142]	https://www.ipd.kit.edu/mitarbeiter/muellere/consub/
	Amazon-v1	1	314K	882K	28	6.2K	[8], [26], [71], [72], [75], [76], [102]	https://www.ipd.kit.edu/mitarbeiter/muellere/consub/
	Amazon-v2	1	11K	-	25	821	-	https://github.com/dmlc/dgl/blob/master/python/dgl/data/fraud.py
	Elliptic	1	203K	234K	166	4.5K	-	https://www.kaggle.com/ellipticco/elliptic-data-set
	Yelp	1	45K	-	32	6.6K	-	https://github.com/dmlc/dgl/blob/master/python/dgl/data/fraud.py
Transportation Networks	New York City Taxi	-	-	-	-	-	[39], [110], [132]	http://www.nyc.gov/html/tlc/html/about/triprecorddata.shtml

* -: Not Given, #G: Number of Graphs, #N: Number of Nodes, #E: Number of Edges, #FT: Number of Features, #AN: Number of Anomalies, REF: References.

TABLE 6: Evaluation Metrics. tp: true positives; tn: true negatives; fp: false positives; fn: false negatives.

Evaluation Metric	Formula/Description
Accuracy	$\frac{tp+tn}{tp+tn+fp+fn}$
Precision	$\frac{tp}{tp+fp}$
Recall	$\frac{tp}{tp+fn}$
F1 Score	$2 * \frac{Recall*Precision}{Recall+Precision}$
AUC-ROC	The Area Under the ROC Curve
AUC-AP	The Area Under Precision-Recall Curve

normal objects and anomalies in particular applications. Typically, this involves extracting edge/sub-graph/graph-level features that can capture these clues from both structure and often node/edge attributes, modeling the patterns of normal/abnormal edges/sub-graphs/graphs using these features, and measuring the abnormalities accordingly. However, current deep learning based graph anomaly detection techniques put very few efforts on these.

Opportunities: We believe more research efforts should be put on anomalous edge, sub-graph, and graph detection with regard to their significance in real-world applications. All possible solutions should first consider the application domain and explore domain knowledge to find complementary clues as basis for these problems. Then, motivated by recent advances in deep learning for edge, sub-graph, and graph-level representation learning [120], [168], extensive works can be done to learn an anomaly aware embedding space such that abnormal patterns of anomalies are feasible to extract. Although this direction seems quite straightforward, the true challenges are lying behind the specific application domains, hence domain knowledge, anomalous pattern recognition and anomaly aware deep learning techniques should be enforced simultaneously.

10.2 Anomaly Detection in Dynamic Graphs

Dynamic graphs provide a powerful machinery to capture the evolving relationships between real objects and their attributes. The ever-changing structure and attribute information inherently makes anomaly detection very challenging and leads to two primary concerns for the task. The first is to consider the spatial and temporal information contained in each graph snapshot at different time stamps and the second is to explore the evolving patterns of nodes, edges, sub-graphs and graphs as well as their interaction with the node/edge attributes over time. When these challenges are maturely tackled, the detection techniques will achieve better results.

Opportunities: From our observation, most of deep learning based dynamic graph anomaly detection techniques are built on DeepWalk [78], GCN [89] or other deep models that are intuitively designed for static graphs. Nevertheless, other information (e.g., the attribute evolving patterns [169], [170]) are utilized inadequately for the detection task. Therefore, we identify the following directions in which future works can be targeted on.

- *Utilizing dynamic graph mining tools.* As a popular research topic, deep learning for dynamic graph data mining [171], [172] has shown its effectiveness in supporting many dynamic graph analysis tasks, such as link prediction, node classification. More future works can be foreseen in adopting these techniques for anomaly detection.
- *Deriving solid evidence for anomaly detection.* The affluent structural, attribute and temporal information in dynamic graphs are rich sources for anomaly detection. Apart from the evidences widely used in current works (e.g., burst of connections between node pairs, suddenly vanishing connections), we suggest to explore structure changes and attribute changes in depth and derive additional information, such as the appearance of abnormal attributes, to enhance the detection performance.
- *Handling complex dynamics.* Real-world networks always exhibit changes in both the network structure and node attributes, but, only very few works aim to handle this scenario. Most of the state-of-arts only consider the changes from one aspect. Although this scenario is extremely complex and detecting anomalies in this kind of dynamic graphs is very challenging, it is worth studying because of the representativeness of these graphs in reflecting real-network data.

10.3 Anomaly Detection in Heterogeneous Graphs

Heterogeneous graphs are a specific type of graphs that contain different types of nodes and edges. For instance, Twitter can be intuitively modeled as a heterogeneous graph which might be comprised of tweets (source tweets/retweets), users, words, etc.

Opportunities: In order to utilize the complex relationships between different types of nodes in heterogeneous graphs for anomaly detection, representative works, such as HGATRD [146], GCAN [145] and GLAN [147], typically decompose the heterogeneous graph into individual graphs (e.g., one with tweets and users, and another with tweets and words) according to meta-paths and then adopt D(G)NNs to learn the embeddings for graph anomaly detection. Such a decomposition inherently overlooks the direct relations between different types of nodes/edges and downgrades the effectiveness of the embeddings. A possible solution is to reveal the complex relations between different types of nodes and edges, and encoding them into a unique representation for boosted detection performance.

10.4 Anomaly Detection in Large-scale Graphs

The scalability to high-dimensional and large-scale data is an ongoing and significant challenge to anomaly detection techniques. In face of large-scale networks, such as online social networks (e.g., Facebook, Twitter) that contain billions of users and friendship links, the data size (in terms of both graph size and number of node attributes) will be extremely high. However, most of the existing works lack the capability to detect anomalies in such large-scale data because they are transductive models and need to take the whole graph as input for further analysis. The computation time and memory cost will increase dramatically as the network scales up and this prevents the adoption of existing techniques on large-scale networks.

Opportunities: Accordingly, there is a need for scalable graph anomaly detection techniques. A possible approach would be inductive learning that first trains a detection model on part of the whole graph and then applies the model to detect anomalies in the unseen data. As inductive learning models, such as GraphSAGE [173], have shown their effectiveness on link prediction and node classification in large-scale graphs, this approach is expected to provide basis for graph anomaly detection in large-scale graphs and similar techniques can be investigated in the future.

10.5 Multi-view Graph Anomaly Detection

In real-world networks, objects might form different kinds of relationships with others (e.g., user's followership and friendship on Twiter) and their attribute information might be collected from different resources (e.g., user's profile, historical posts). This results in two types of multi-view graphs: 1) multi-graph that contains more than one type of edges between two nodes [174], [175]; and 2) multi-attributed-view graph that stores node attributes in different attributed views [91], [176], [177].

Opportunities: These multi-views basically allow us to analyze real objects' characteristics from different perspectives. Each view also provides complementary information to other views and they might have different significance on anomaly detection. For instance, anomalies might be indistinguishable in one view but are obviously divergent from the majority in another view. These exist a variety of work in data mining on multi-aspect learning [178]–[180], however, work that can accommodate multi-view graphs along with multi-view attributes on nodes for anomaly detection purposes is nascent. The affluent information contained in multiple views and the inconsistency among them are overlooked in these works. To this end, we believe more research effort in this direction is needed and digesting the relationships between views is vital to the success as two views might provide contrary/supplementary information for anomaly detection.

10.6 Camouflaged/Adversarial Anomaly Detection

The easy accessibility of online platforms/systems has made them convenient platforms for fraudsters, attacker and other malevolent agents to carry out malicious activities and obtain unexpected benefits. Although various anomaly detection systems have been deployed to protect benign objects, such as benign users in online social networks and normal commodities in online review networks, anomalies are concealing themselves to evade the detections as well [181]. This is widely recognized as camouflaged anomalies, which typically disguise themselves as regular objects. If the detection techniques are not robust against such cases, or otherwise can effectively and quickly adapt to the evolving behavior of evasion-seeking attackers, the anomalies can cause huge losses to the society as well as businesses.

Opportunities: In face of camouflage, the boundary between anomalies and regular objects will be blurred, making anomalies much harder to be identified. We believe extensive efforts should be put on detecting these anomalies because very few works have highlighted their capability in handling camouflaged anomalies in graphs [26], [27], [75]. To fulfill the gap, one major direction is to jointly analyze the co-relations (such as the triadic, tetradic, or high-order relationships between objects in hypergraphs [182]–[185]), attribute and other information, comprised in graphs. By this, anomalies that only camouflage their local structures or attributes can be identified effectively. Enhancing existing techniques can be another direction. This involves incorporating additional detection mechanisms or function blocks particularly designed for distinguishing camouflaged anomalies with existing detection techniques. Consequently, these techniques will bridge most existing works and camouflaged anomaly detection.

10.7 Imbalanced Graph Anomaly Detection

For the rare occurrences of anomalies, the number of anomalous objects is naturally far less than the regulars, introducing imbalanced class (i.e., anomalies and non-anomalies) distributions in the training data. As deep learning models rely heavily on the training data, such imbalance will pose great challenges to graph anomaly detection and it remains a significant obstacle to deep learning techniques. Typically, the imbalanced class distributions will downgrade the detection techniques' capability of capturing the difference between anomalies and non-anomalies, and it might even cause over-fitting on the anomalies class because the number of anomalies is too small in the training data. If the detection model overlooks this critical fact and is trained improperly, its detection performance (e.g., accuracy, precision) will be sub-optimal.

Opportunities: In fact, class imbalance has been widely studied in many research areas [186], [187]. Their advances shed important light (e.g., undersample majority class, modify the algorithms) on solving the imbalanced training problem, but the contemporary graph anomaly detection methods rarely incorporate these techniques. For more effective detection techniques, biased models that pay more attention to anomalies, such as penalizing additional training losses on misclassified anomalies, would be a possible direction to circumvent the problem. Moreover, when adopting graph neural networks (e.g., GCN, GraphSAGE) that aggregate neighboring information to the target node in future works, the over-smoothing between connected nodes' features should be prevented such that distinguishable features of the minority class (anomalies) can be preserved to support anomaly detection.

10.8 Multi-task Anomaly Detection

Graph anomaly detection has close relations with other graph mining tasks such as community detection [57], node classification [188], and link prediction [189]. For a concrete example, when detecting community anomalies, community detection techniques are usually applied to extract the community structure prior to anomaly detection. Meanwhile, the anomaly detection results can be utilized to optimize the community structure. Such mutually beneficial collaborations between anomaly detection and other tasks inherently suggest opportunity for multi-task learning that can handle diverse tasks simultaneously and share information between them.

Opportunities: Multi-task learning provides an effective machinery to incorporate associated tasks [190], [191]. For anomaly detection, its utmost advantage is that the training signal from another task yields complementary information to distinguish anomalies and non-anomalies, leading to enhanced detection performance. However, very few attempts focuses on this at present. Apart from current works, such as [97] that jointly performs anomalous node detection and personalized recommendation, explorations on combining other learning tasks with graph anomaly detection is likely to emerge as a fruitful future direction.

10.9 Graph Anomaly Interpretability

The interpretability of anomaly detection techniques is vital to subsequent anomaly handling process. When applying these techniques to real applications such as financial and insurance systems, it is a prerequisite to provide explainable and lawful evidence to support the detection results. However, most of the existing works lack the capability to provide such evidence. To identify the anomalies, the most commonly used metrics are top-k ranking and simple anomaly scoring functions. These metrics are flexible to label objects as anomalies or non-anomalies, but they cannot derive solid explanations. Moreover, as deep learning techniques have also been criticized for their low interpretability, future works on graph anomaly detection with deep learning should pay much more attention to this [192].

Opportunities: In order to bridge the gap, integrating specially designed interpretation algorithms or mechanisms [193], [194] into the detection framework would be a possible solution but this will inherently introduce more computational cost. Future works should therefore balance the cost of anomaly detection performance and interpretability. Visualization-based approaches (e.g., dashboards, charts) might also be feasible to apply to show the distinction between anomalies and non-anomalies in a human friendly manner. Further research in this direction will be successful if interpretable visualization results can be given [195].

10.10 Graph Anomaly Identification Strategies

Amongst existing unsupervised graph anomaly detection techniques, anomalies are mainly identified based on residual analysis [41], [102], reconstruction loss [104], distance-based statistics [83], density-based statistics [95], and one-class classification [109]. The underlying intuition of these identification strategies is that anomalies have inconsistent data patterns with regular objects and they will therefore: 1) introduce more residual errors or are harder to reconstruct, or 2) locate in low-density areas or far away from the majorities in an anomaly-aware feature space which is learned by the detection methods. Effort toward designing novel loss functions for GNNs for anomaly detection is currently quite limited [66]

Opportunities: Although these strategies could capture the deviating data patterns of anomalies, they also have different limitations. Specifically, the residual analysis, one-class classification and reconstruction loss strategies are sensitive to noisy training data. The noisy nodes, edges or sub-graphs would also exhibit large residuals, large distance to origin/hypersphere center and high reconstruction losses. Meanwhile, the distance-based and density-based strategies can only be applied when anomalies and non-anomalies are well separated in the lower-dimensional space. The detection performance will also downgrade dramatically if the gap between anomalies and non-anomalies is not that evident (e.g. under camouflage). This calls for extensive future efforts to break these limitations and explore the anomaly identification strategies.

10.11 Systematic Benchmarking

A systematic benchmarking is key to evaluate the performance of graph anomaly detection techniques. As indicated from our analysis in Section 9.4, recent studies have raised continuous attention on more comprehensive and effective benchmarking [135], [165]–[167]. Typically, the benchmarking framework consists benchmark datasets, baseline methods, evaluation metrics, and further analysis tools. When evaluating the techniques' performance with other baselines, the evaluation dataset and metrics become very important because the performance of each model may vary depending on the setting. The shortage of public datasets and (public available) baseline methods also imposes great challenges for effective evaluation. Although one of the aims of our survey is to provide extensive materials (i.e., open-sourced implementations, datasets, evaluation metrics), this work can only serve as a basis for future works toward a systematic benchmarking. We invite more efforts from the anomaly detection community to boost this important aspect. Certainly, rigorous attention to designing better benchmark datasets for evaluation would help reveal the strengths and limitations of various detection models, and ultimately keep a fair and accurate record of progress in this field.

10.12 Unified Anomaly Detection Framework

Graph anomalies can be categorized as anomalous node, edge, and sub-graph in a single graph and graph anomalies in a graph database. These anomalies usually coexist in real-world datasets. For instance, individual fraudsters, abnormal relationships, and fraud groups exist concurrently in online social networks (as shown in Fig. 1). Moreover, there may be different ways to define anomalies of certain type, such as community outliers versus anomalous communities or attribute-based versus structural node anomalies. When deploying detection techniques in real applications, it is expected that all types of anomalies can be identified while consuming the least resources and time. A straightforward approach is an integration of independent anomalous node, edge, and sub-graph detection techniques. Although this is convenient to apply to relatively small networks, its high computational cost will severely prevent the approach from scaling to large networks, such as Facebook and Twitter, because the same graph data has to be loaded and processed more than once by different techniques.

Opportunities: Unified frameworks that can detect diverse types of anomalies together [196], [197] provide feasible solutions to bridge the gap. To build such frameworks, one possible direction is to capture all the information needed by different detection techniques simultaneously so that these techniques can be applied. The idea seems to be non-challenging, but in deep learning, how to design neural network layers and learning strategies that can fulfill this need requires extensive efforts.

11 Conclusion

Due to the complex relationships between real-world objects and recent advances in deep learning (especially graph neural networks), graph anomaly detection with deep learning is currently at the forefront of anomaly detection. To the best of our knowledge, this is the first survey that presents a comprehensive review dedicated to graph anomaly detection with modern deep learning techniques. Specifically, we have reviewed and categorized contemporary DL techniques according to the types of graph anomalies they can detect as: (1) anomalous node detection; (2) anomalous edge detection; (3) anomalous sub-graph detection and finally, (4) anomalous graph 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 basis for systematic benchmarking by compiling a wide-range of commonly used datasets, open-sourced implementations and synthetic dataset generation techniques, and further highlight twelve potential directions for future works according to our survey results. It is our firm belief that graph anomaly detection with deep learning is indeed more than a burst of temporary interest, and that numerous applications from diverse domain is to surely benefit from it for the years to come.

References

[1] F. E. Grubbs, "Procedures for detecting outlying observations in samples," *Technometrics*, vol. 11, no. 1, pp. 1–21, 1969.

[2] B. Hooi, N. Shah, A. Beutel, S. Günnemann, L. Akoglu, M. Kumar, D. Makhija, and C. Faloutsos, "Birdnest: Bayesian inference for ratings-fraud detection," in *SDM*, 2016, pp. 495–503.

[3] K. Hinkelmann, S. Ahmed, and F. Corradini, "Combining machine learning with knowledge engineering to detect fake news in social networks - a survey," in *AAAI*, vol. 2350, 2019.

[4] V. Nguyen, K. Sugiyama, P. Nakov, and M. Kan, "Fang: Leveraging social context for fake news detection using graph representation," in *CIKM*, 2020, pp. 1165–1174.

[5] N. T. Tam, M. Weidlich, B. Zheng, H. Yin, N. Q. V. Hung, and B. Stantic, "From anomaly detection to rumour detection using data streams of social platforms," *VLDB J.*, vol. 12, no. 9, pp. 1016–1029, 2019.

[6] R. Yu, H. Qiu, Z. Wen, C. Lin, and Y. Liu, "A survey on social media anomaly detection," *SIGKDD Explor.*, vol. 18, no. 1, pp. 1–14, 2016.

[7] A. Benamira, B. Devillers, E. Lesot, A. K. Ray, M. Saadi, and F. D. Malliaros, "Semi-supervised learning and graph neural networks for fake news detection," in *ASONAM*, 2019, pp. 568–569.

[8] S. Kumar, B. Hooi, D. Makhija, M. Kumar, C. Faloutsos, and V. S. Subrahmanian, "Rev2: Fraudulent user prediction in rating platforms," in *WSDM*, 2018, pp. 333–341.

[9] P. Bogdanov, C. Faloutsos, M. Mongiovì, E. E. Papalexakis, R. Ranca, and A. K. Singh, "Netspot: Spotting significant anomalous regions on dynamic networks," in *SDM*, 2013, pp. 28–36.

[10] B. A. Miller, N. Arcolano, and N. T. Bliss, "Efficient anomaly detection in dynamic, attributed graphs: Emerging phenomena and big data," in *ISI*, 2013, pp. 179–184.

[11] K. Miao, X. Shi, and W. Zhang, "Attack signal estimation for intrusion detection in industrial control system," *Comput. Secur.*, vol. 96, p. 101926, 2020.

[12] B. Perozzi and L. Akoglu, "Scalable anomaly ranking of attributed neighborhoods," in *SDM*, 2016, pp. 207–215.

[13] G. Zhang, Z. Li, J. Huang, J. Wu, C. Zhou, and J. Yang, "efraudcom: An e-commerce fraud detection system via competitive graph neural networks," *ACM Trans. Inf. Syst.*, 2018.

[14] B. Branco, P. Abreu, A. S. Gomes, M. S. C. Almeida, J. T. Ascensão, and P. Bizarro, "Interleaved sequence rnns for fraud detection," in *KDD*, 2020, pp. 3101–3109.

[15] C. Liu, Q. Zhong, X. Ao, L. Sun, W. Lin, J. Feng, Q. He, and J. Tang, "Fraud transactions detection via behavior tree with local intention calibration," in *KDD*, 2020, pp. 3035–3043.

[16] Q. Guo, Z. Li, B. An, P. Hui, J. Huang, L. Zhang, and M. Zhao, "Securing the deep fraud detector in large-scale e-commerce platform via adversarial machine learning approach," in *WWW*, 2019, pp. 616–626.

[17] B. Iglewicz and D. C. Hoaglin, *How to detect and handle outliers*. Asq Press, 1993, vol. 16.

[18] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM Comput. Surv.*, vol. 41, no. 3, pp. 15:1–15:58, 2009.

[19] T. Pourhabibi, O. Kok-Leong, B. H. Kam, and B. Y. Ling, "Fraud detection: A systematic literature review of graph-based anomaly detection approaches," *Decis. Support Syst.*, p. 113303, 2020.

[20] X. Sun, C. Zhang, G. Li, D. Sun, F. Ren, A. Y. Zomaya, and R. Ranjan, "Detecting users' anomalous emotion using social media for business intelligence," *J. Comput. Sci.*, vol. 25, pp. 193–200, 2018.

[21] J. Wu, Z. Hong, S. Pan, X. Zhu, Z. Cai, and C. Zhang, "Multi-graph-view learning for graph classification," in *ICDM*, 2014, pp. 590–599.

[22] Z. Wang and C. Lan, "Towards a hierarchical bayesian model of multi-view anomaly detection," in *IJCAI*, 2020, pp. 2420–2426.

[23] G. Pang, L. Cao, L. Chen, and H. Liu, "Learning representations of ultrahigh-dimensional data for random distance-based outlier detection," in *KDD*, 2018, pp. 2041–2050.

[24] G. Pang, C. Shen, L. Cao, and A. V. D. Hengel, "Deep learning for anomaly detection," *ACM Comput. Surv.*, vol. 54, no. 2, p. 1–38, 2021.

[25] L. Akoglu, H. Tong, and D. Koutra, "Graph based anomaly detection and description: A survey," *Data Min. Knowl. Discovery*, vol. 29, no. 3, pp. 626–688, 2015.

[26] B. Hooi, K. Shin, H. A. Song, A. Beutel, N. Shah, and C. Faloutsos, "Graph-based fraud detection in the face of camouflage," *ACM Trans. Knowl. Discovery Data*, vol. 11, no. 4, pp. 44:1–44:26, 2017.

[27] Y. Dou, Z. Liu, L. Sun, Y. Deng, H. Peng, and P. S. Yu, "Enhancing graph neural network-based fraud detectors against camouflaged fraudsters," in *CIKM*, 2020, pp. 315–324.

[28] S. Pandit, D. H. Chau, S. Wang, and C. Faloutsos, “Netprobe: fast and scalable system for fraud detection in online auction networks,” in *WWW*, 2007, pp. 201–210.

[29] K. Shin, B. Hooi, J. Kim, and C. Faloutsos, “Densealert: Incremental dense-subtensor detection in tensor streams,” in *KDD*, 2017, pp. 1057–1066.

[30] Z. Liu, J. X. Yu, Y. Ke, X. Lin, and L. Chen, “Spotting significant changing subgraphs in evolving graphs,” in *ICDM*, 2008, pp. 917–922.

[31] J. Wu, Z. Hong, S. Pan, X. Zhu, C. Zhang, and Z. Cai, “Multi-graph learning with positive and unlabeled bags,” in *SDM*, 2014, pp. 217–225.

[32] L. Gao, J. Wu, C. Zhou, and Y. Hu, “Collaborative dynamic sparse topic regression with user profile evolution for item recommendation,” in *AAAI*, 2017, pp. 1316—1322.

[33] C. C. Aggarwal, Y. Zhao, and P. S. Yu, “Outlier detection in graph streams,” in *ICDE*, 2011, pp. 399–409.

[34] J. Wu, S. Pan, X. Zhu, C. Zhang, and P. S. Yu, “Multiple structure-view learning for graph classification,” *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 7, pp. 3236–3251, 2018.

[35] H. Wang, C. Zhou, X. Chen, J. Wu, S. Pan, and J. Wang, “Graph stochastic neural networks for semi-supervised learning,” in *NIPS*, 2020.

[36] Z. Chen, W. Hendrix, and N. F. Samatova, “Community-based anomaly detection in evolutionary networks,” *J. Intell. Inf. Syst.*, vol. 39, no. 1, pp. 59–85, 2012.

[37] F. Jie, C. Wang, F. Chen, L. Li, and X. Wu, “Block-structured optimization for anomalous pattern detection in interdependent networks,” in *ICDM*, 2019, pp. 1138–1143.

[38] L. Akoglu, M. McGlohon, and C. Faloutsos, “Oddball: Spotting anomalies in weighted graphs,” in *PAKDD*, 2010, pp. 410–421.

[39] D. Eswaran, C. Faloutsos, S. Guha, and N. Mishra, “Spotlight: Detecting anomalies in streaming graphs,” in *KDD*, 2018, pp. 1378–1386.

[40] N. Li, H. Sun, K. C. Chipman, J. George, and X. Yan, “A probabilistic approach to uncovering attributed graph anomalies,” in *SDM*, 2014, pp. 82–90.

[41] J. Li, H. Dani, X. Hu, and H. Liu, “Radar: Residual analysis for anomaly detection in attributed networks,” in *IJCAI*, 2017, pp. 2152–2158.

[42] S. M. Erfani, S. Rajasegarar, S. Karunasekera, and C. Leckie, “High-dimensional and large-scale anomaly detection using a linear one-class SVM with deep learning,” *Pattern Recognit.*, vol. 58, pp. 121–134, 2016.

[43] S. Thudumu, P. Branch, J. Jin, and J. J. Singh, “A comprehensive survey of anomaly detection techniques for high dimensional big data,” *J. Big Data*, vol. 7, no. 1, p. 42, 2020.

[44] A. Boukerche, L. Zheng, and O. Alfandi, “Outlier detection: Methods, models, and classification,” *ACM Comput. Surv.*, vol. 53, no. 3, pp. 55:1–55:37, 2020.

[45] S. Bulusu, B. Kailkhura, B. Li, P. K. Varshney, and D. Song, “Anomalous instance detection in deep learning: A survey,” *arXiv preprint arXiv:2003.06979*, 2020.

[46] R. Chalapathy and S. Chawla, “Deep learning for anomaly detection: A survey,” *arXiv preprint arXiv:1901.03407*, 2019.

[47] S. Ranshous, S. Shen, D. Koutra, S. Harenberg, C. Faloutsos, and N. F. Samatova, “Anomaly detection in dynamic networks: A survey,” *Wiley Interdiscip. Rev. Comput. Stat.*, vol. 7, no. 3, pp. 223–247, 2015.

[48] D. J. D’Souza and K. U. K. Reddy, “Anomaly detection for big data using efficient techniques: A review,” *AIDE*, pp. 1067–1080, 2021.

[49] S. Eltanbouly, M. Bashendy, N. AlNaimi, Z. Chkirbene, and A. Erbad, “Machine learning techniques for network anomaly detection: A survey,” in *ICIoT*, 2020, pp. 156–162.

[50] G. Fernandes, J. J. Rodrigues, L. F. Carvalho, J. F. Al-Muhtadi, and M. L. Proença, “A comprehensive survey on network anomaly detection,” *Telecommun. Syst.*, vol. 70, no. 3, pp. 447–489, 2019.

[51] D. Kwon, H. Kim, J. Kim, S. C. Suh, I. Kim, and K. J. Kim, “A survey of deep learning-based network anomaly detection,” *Clust. Comput.*, vol. 22, pp. 949–961, 2019.

[52] P. Gogoi, D. K. Bhattacharyya, B. Borah, and J. K. Kalita, “A survey of outlier detection methods in network anomaly identification,” *Comput. J.*, vol. 54, no. 4, pp. 570–588, 2011.

[53] D. Savage, X. Zhang, X. Yu, P. Chou, and Q. Wang, “Anomaly detection in online social networks,” *Soc. Networks*, vol. 39, pp. 62–70, 2014.

[54] R. Wang, K. Nie, T. Wang, Y. Yang, and B. Long, “Deep learning for anomaly detection,” in *WSDM*, 2020, pp. 894–896.

[55] Y. Zhang, J. Wu, Z. Cai, B. Du, and S. Y. Philip, “An unsupervised parameter learning model for rvfl neural network,” *Neural Networks*, vol. 112, pp. 85–97, 2019.

[56] Q. Wang, W. Zhao, J. Yang, J. Wu, C. Zhou, and Q. Xing, “Atne-trust: Attributed trust network embedding for trust prediction in online social networks,” in *ICDM*, 2020, pp. 601–610.

[57] F. Liu, S. Xue, J. Wu, C. Zhou, W. Hu, C. Paris, S. Nepal, J. Yang, and P. S. Yu, “Deep learning for community detection: progress, challenges and opportunities,” in *IJCAI*, 2020, pp. 4981–4987.

[58] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and P. S. Yu, “A comprehensive survey on graph neural networks,” *IEEE Trans. Neural Networks Learn. Syst.*, vol. 32, no. 1, pp. 4–24, 2021.

[59] P. Cui, X. Wang, J. Pei, and W. Zhu, “A survey on network embedding,” *IEEE Trans. Knowl. Data Eng.*, vol. 31, no. 5, pp. 833–852, 2019.

[60] S. Zhu, S. Pan, C. Zhou, J. Wu, Y. Cao, and B. Wang, “Graph geometry interaction learning,” in *NIPS*, 2020.

[61] X. Su, S. Xue, F. Liu, J. Wu, J. Yang, C. Zhou, W. Hu, C. Paris, S. Nepal, D. Jin *et al.*, “A comprehensive survey on community detection with deep learning,” *arXiv preprint arXiv:2105.12584*, 2021.

[62] C. C. Noble and D. J. Cook, “Graph-based anomaly detection,” in *KDD*, 2003, pp. 631–636.

[63] X. Teng, M. Yan, A. M. Ertugrul, and Y. Lin, “Deep into hypersphere: Robust and unsupervised anomaly discovery in dynamic networks,” in *IJCAI*, 2018, pp. 2724–2730.

[64] N. Shah, A. Beutel, B. Hooi, L. Akoglu, S. Günnemann, D. Makhija, M. Kumar, and C. Faloutsos, “Edgecentric: Anomaly detection in edge-attributed networks,” in *ICDM*, 2016, pp. 327–334.

[65] B. Wang, N. Z. Gong, and H. Fu, “Gang: Detecting fraudulent users in online social networks via guilt-by-association on directed graphs,” in *ICDM*, 2017, pp. 465–474.

[66] T. Zhao, C. Deng, K. Yu, T. Jiang, D. Wang, and M. Jiang, “Error-bounded graph anomaly loss for gnns,” in *CIKM*, 2020, pp. 1873–1882.

[67] Q. Zhang and S. Zhu, “Visual interpretability for deep learning: a survey,” *Frontiers Inf. Technol. Electron. Eng.*, vol. 19, no. 1, pp. 27–39, 2018.

[68] L. Akoglu, “Anomaly mining–past, present and future,” *arXiv preprint arXiv:2105.10077*, 2021.

[69] Y. Zhao, R. A. Rossi, and L. Akoglu, “Automating outlier detection via meta-learning,” *arXiv preprint arXiv:2009.10606*, 2020.

[70] L. Ruff, J. R. Kauffmann, R. A. Vandermeulen, G. Montavon, W. Samek, M. Kloft, T. G. Dietterich, and K.-R. Müller, “A unifying review of deep and shallow anomaly detection,” *IEEE*, 2021.

[71] A. Bojchevski and S. Günnemann, “Bayesian robust attributed graph clustering: Joint learning of partial anomalies and group structure,” in *AAAI*, 2018, pp. 2738–2745.

[72] M. Zhu and H. Zhu, “Mixedad: A scalable algorithm for detecting mixed anomalies in attributed graphs,” in *AAAI*, 2020, pp. 1274–1281.

[73] B. Perozzi, L. Akoglu, P. I. Sánchez, and E. Müller, “Focused clustering and outlier detection in large attributed graphs,” in *KDD*, 2014, pp. 1346–1355.

[74] Q. Ding, N. Katenka, P. Barford, E. D. Kolaczyk, and M. Crovella, “Intrusion as (anti)social communication: characterization and detection,” in *KDD*, 2012, pp. 886–894.

[75] B. Hooi, H. A. Song, A. Beutel, N. Shah, K. Shin, and C. Faloutsos, “Fraudar: Bounding graph fraud in the face of camouflage,” in *KDD*, 2016, pp. 895–904.

[76] R. Hu, C. C. Aggarwal, S. Ma, and J. Huai, “An embedding approach to anomaly detection,” in *ICDE*, 2016, pp. 385–396.

[77] G. Karypis and V. Kumar, “A fast and high quality multilevel scheme for partitioning irregular graphs,” *J. Sci. Comput.*, vol. 20, no. 1, pp. 359–392, 1998.

[78] B. Perozzi, R. Al-Rfou, and S. Skiena, “Deepwalk: Online learning of social representations,” in *KDD*, 2014, pp. 701–710.

[79] A. Grover and J. Leskovec, “Node2vec: Scalable feature learning for networks,” in *KDD*, 2016, pp. 855–864.

[80] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei, “Line: Large-scale information network embedding,” in *WWW*, 2015, pp. 1067–1077.

[81] S. Bandyopadhyay, L. N, S. V. Vivek, and M. N. Murty, “Outlier resistant unsupervised deep architectures for attributed network embedding,” in *WSDM*, 2020, pp. 25–33.

[82] S. Bandyopadhyay, N. Lokesh, and M. N. Murty, “Outlier aware network embedding for attributed networks,” in *AAAI*, 2019, pp. 12–19.

[83] W. Yu, W. Cheng, C. C. Aggarwal, K. Zhang, H. Chen, and W. Wang, “Netwalk: A flexible deep embedding approach for anomaly detection in dynamic networks,” in *KDD*, 2018, pp. 2672–2681.

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

[85] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander, “Lof: identifying density-based local outliers,” in *SIGMOD*, 2000, pp. 93–104.

[86] C. C. Aggarwal and P. S. Yu, “Outlier detection for high dimensional data,” in *SIGMOD*, 2001, pp. 37–46.

[87] W. L. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *NIPS*, 2017, pp. 1024–1034.

[88] W. L. Hamilton, R. Ying, and J. Leskovec, "Representation learning on graphs: Methods and applications," *IEEE Data Eng. Bull.*, vol. 40, no. 3, pp. 52–74, 2017.

[89] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *ICLR*, 2017.

[90] K. Ding, J. Li, R. Bhanushali, and H. Liu, "Deep anomaly detection on attributed networks," in *SDM*, 2019, pp. 594–602.

[91] Z. Peng, M. Luo, J. Li, L. Xue, and Q. Zheng, "A deep multi-view framework for anomaly detection on attributed networks," *IEEE Trans. Knowl. Data Eng.*, 2020.

[92] X. Sheng, D. Zhan, S. Lu, and Y. Jiang, "Multi-view anomaly detection: neighborhood in locality matters," in *AAAI*, 2019, pp. 4894–4901.

[93] J. Wu, S. Pan, X. Zhu, and Z. Cai, "Boosting for multi-graph classification," *IEEE Trans. Cybern.*, vol. 45, no. 3, pp. 416–429, 2015.

[94] J. Wu, X. Zhu, C. Zhang, and Z. Cai, "Multi-instance multi-graph dual embedding learning," in *ICDM*, 2013, pp. 827–836.

[95] Y. Li, X. Huang, J. Li, M. Du, and N. Zou, "Specae: Spectral autoencoder for anomaly detection in attributed networks," in *CIKM*, 2019, pp. 2233–2236.

[96] J. Wang, R. Wen, C. Wu, Y. Huang, and J. Xion, "Fdgars: Fraudster detection via graph convolutional networks in online app review system," in *WWW*, 2019, pp. 310–316.

[97] S. Zhang, H. Yin, T. Chen, Q. V. H. Nguyen, Z. Huang, and L. Cui, "Gcn-based user representation learning for unifying robust recommendation and fraudster detection," in *SIGIR*, 2020, pp. 689–698.

[98] K. Ding, J. Li, and H. Liu, "Interactive anomaly detection on attributed networks," in *WSDM*, 2019, pp. 357–365.

[99] J. Langford and T. Zhang, "The epoch-greedy algorithm for multi-armed bandits with side information," in *NIPS*, 2008, pp. 817–824.

[100] P. Morales, R. S. Caceres, and T. Eliassi-Rad, "Selective network discovery via deep reinforcement learning on embedded spaces," *Appl. Network Sci.*, vol. 6, no. 1, pp. 1–20, 2021.

[101] N. Liu, X. Huang, and X. Hu, "Accelerated local anomaly detection via resolving attributed networks," in *IJCAI*, 2017, pp. 2337–2343.

[102] Z. Peng, M. Luo, J. Li, H. Liu, and Q. Zheng, "Anomalous: A joint modeling approach for anomaly detection on attributed networks," in *IJCAI*, 2018, pp. 3513–3519.

[103] L. Wu, X. Hu, F. Morstatter, and H. Liu, "Adaptive spammer detection with sparse group modeling," in *ICWSM*, 2017, pp. 319–326.

[104] H. Fan, F. Zhang, and Z. Li, "Anomalydae: Dual autoencoder for anomaly detection on attributed networks," in *ICASSP*, 2020, pp. 5685–5689.

[105] D. Wang, Y. Qi, J. Lin, P. Cui, Q. Jia, Z. Wang, Y. Fang, Q. Yu, J. Zhou, and S. Yang, "A semi-supervised graph attentive network for financial fraud detection," in *ICDM*, 2019, pp. 598–607.

[106] K. Ding, J. Li, N. Agarwal, and H. Liu, "Inductive anomaly detection on attributed networks," in *IJCAI*, 2020, pp. 1288–1294.

[107] L. Zhang, J. Yuan, Z. Liu, Y. Pei, and L. Wang, "A robust embedding method for anomaly detection on attributed networks," in *IJCNN*, 2019, pp. 1–8.

[108] J. Liang, P. Jacobs, J. Sun, and S. Parthasarathy, "Semi-supervised embedding in attributed networks with outliers," in *SDM*, 2018, pp. 153–161.

[109] X. Wang, Y. Du, P. Cui, and Y. Yang, "Ocgnn: One-class classification with graph neural networks," *arXiv preprint arXiv:2002.09594*, 2020.

[110] X. Teng, Y. Lin, and X. Wen, "Anomaly detection in dynamic networks using multi-view time-series hypersphere learning," in *CIKM*, 2017, pp. 827–836.

[111] P. Zheng, S. Yuan, X. Wu, J. Li, and A. Lu, "One-class adversarial nets for fraud detection," in *AAAI*, 2019, pp. 1286–1293.

[112] H. Wang, J. Wu, W. Hu, and X. Wu, "Detecting and assessing anomalous evolutionary behaviors of nodes in evolving social networks," *ACM Trans. Knowl. Discovery Data*, vol. 13, no. 1, pp. 12:1–12:24, 2019.

[113] N. Ailon, R. Jaiswal, and C. Monteleoni, "Streaming k-means approximation." in *NIPS*, vol. 4, 2009, p. 2.

[114] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. C. Courville, and Y. Bengio, "Generative adversarial nets," in *NIPS*, 2014.

[115] Z. Dai, Z. Yang, F. Yang, W. W. Cohen, and R. Salakhutdinov, "Good semi-supervised learning that requires a bad gan," in *NIPS*, 2017.

[116] N. Srivastava, E. Mansimov, and R. Salakhutdinov, "Unsupervised learning of video representations using lstms," in *ICML*, vol. 37, 2015, pp. 843–852.

[117] Y.-Y. Chang, P. Li, R. Sosic, M. Afifi, M. Schweighauser, and J. Leskovec, "F-fade: Frequency factorization for anomaly detection in edge streams," in *WSDM*, 2021, pp. 589–597.

[118] L. Ouyang, Y. Zhang, and Y. Wang, "Unified graph embedding-based anomalous edge detection," in *IJCNN*, 2020, pp. 1–8.

[119] D. Duan, L. Tong, Y. Li, J. Lu, L. Shi, and C. Zhang, "Aane: Anomaly aware network embedding for anomalous link detection," in *ICDM*, 2020, pp. 1002–1007.

[120] L. Xu, X. Wei, J. Cao, and P. S. Yu, "Icane: Interaction content-aware network embedding via co-embedding of nodes and edges," *Int. J. Data Sci. Anal.*, vol. 9, no. 4, pp. 401–414, 2020.

[121] S. Ranshous, S. Harenberg, K. Sharma, and N. F. Samatova, "A scalable approach for outlier detection in edge streams using sketch-based approximations," in *SDM*, 2016, pp. 189–197.

[122] L. Zheng, Z. Li, J. Li, Z. Li, and J. Gao, "Addgraph: Anomaly detection in dynamic graph using attention-based temporal gcn," in *IJCAI*, 2019, pp. 4419–4425.

[123] Q. Cui, S. Wu, Y. Huang, and L. Wang, "A hierarchical contextual attention-based network for sequential recommendation," *Neurocomputing*, vol. 358, pp. 141–149, 2019.

[124] M. Shao, J. Li, F. Chen, and X. Chen, "An efficient framework for detecting evolving anomalous subgraphs in dynamic networks," in *INFOCOM*, 2018, pp. 2258–2266.

[125] H. Wang, C. Zhou, J. Wu, W. Dang, X. Zhu, and J. Wang, "Deep structure learning for fraud detection," in *ICDM*, 2018, pp. 567–576.

[126] M. Ester, H. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *KDD*, 1996, pp. 226–231.

[127] M. Zheng, C. Zhou, J. Wu, S. Pan, J. Shi, and L. Guo, "Fraudne: A joint embedding approach for fraud detection," in *IJCNN*, 2018, pp. 1–8.

[128] J. Wu, X. Zhu, C. Zhang, and P. S. Yu, "Bag constrained structure pattern mining for multi-graph classification," *IEEE Trans. Knowl. Data Eng.*, vol. 26, no. 10, pp. 2382–2396, 2014.

[129] Q. Sun, J. Li, H. Peng, J. Wu, Y. Ning, P. S. Yu, and L. He, "Sugar: Subgraph neural network with reinforcement pooling and self-supervised mutual information mechanism," in *WWW*, 2021, pp. 2081–2091.

[130] E. A. Manzoor, S. M. Milajerdi, and L. Akoglu, "Fast memory-efficient anomaly detection in streaming heterogeneous graphs," in *KDD*, 2016, pp. 1035–1044.

[131] B. Hooi, L. Akoglu, D. Eswaran, A. Pandey, M. Jereminov, L. Pileggi, and C. Faloutsos, "Changedar: Online localized change detection for sensor data on a graph," in *CIKM*, 2018, pp. 507–516.

[132] X. Teng, M. Yan, A. M. Ertugrul, and Y. Lin, "Deep into hypersphere: Robust and unsupervised anomaly discovery in dynamic networks," in *IJCAI*, 2018, pp. 2724–2730.

[133] L. Ruff, R. Vandermeulen, N. Goernitz, L. Deecke, S. A. Siddiqui, A. Binder, E. Müller, and M. Kloft, "Deep one-class classification," in *ICML*, 2018, pp. 4393–4402.

[134] Y. Dou, K. Shu, C. Xia, P. S. Yu, and L. Sun, "User preference-aware fake news detection," *arXiv preprint arXiv:2104.12259*, 2021.

[135] L. Zhao and L. Akoglu, "On using classification datasets to evaluate graph outlier detection: Peculiar observations and new insights," *arXiv preprint arXiv:2012.12931*, 2020.

[136] A. Narayanan, M. Chandramohan, R. Venkatesan, L. Chen, Y. Liu, and S. Jaiswal, "graph2vec: Learning distributed representations of graphs," *arXiv preprint arXiv:1707.05005*, 2017.

[137] S. Verma and Z.-L. Zhang, "Hunt for the unique, stable, sparse and fast feature learning on graphs," in *NIPS*, 2017, pp. 87–97.

[138] Z. Zhang, J. Jia, B. Wang, and N. Z. Gong, "Backdoor attacks to graph neural networks," in *SACMAT*, 2021, pp. 15–26.

[139] H. Dai, H. Li, T. Tian, X. Huang, L. Wang, J. Zhu, and L. Song, "Adversarial attack on graph structured data," in *ICML*, 2018, pp. 1115–1124.

[140] X. Lin, C. Zhou, H. Yang, J. Wu, H. Wang, Y. Cao, and B. Wang, "Exploratory adversarial attacks on graph neural networks," in *ICDM*, 2020, pp. 1136–1141.

[141] D. Eswaran and C. Faloutsos, "Sedanspot: Detecting anomalies in edge streams," in *ICDM*, 2018, pp. 953–958.

[142] L. Gutiérrez-Gómez, A. Bovet, and J. Delvenne, "Multi-scale anomaly detection on attributed networks," in *AAAI*, 2020, pp. 678–685.

[143] H. Nilforoshan and N. Shah, "Slicendice: Mining suspicious multi-attribute entity groups with multi-view graphs," in *DSAA*, 2019, pp. 351–363.

[144] Z. Liu, Y. Dou, P. S. Yu, Y. Deng, and H. Peng, "Alleviating the inconsistency problem of applying graph neural network to fraud detection," in *SIGIR*, 2020, pp. 1569–1572.

[145] Y.-J. Lu and C.-T. Li, "Gcan: Graph-aware co-attention networks for explainable fake news detection on social media," in *ACL*, 2020, pp. 505–514.

[146] Q. Huang, J. Yu, J. Wu, and B. Wang, "Heterogeneous graph attention networks for early detection of rumors on twitter," in *IJCNN*, 2020, pp. 1–8.

[147] C. Yuan, Q. Ma, W. Zhou, J. Han, and S. Hu, "Jointly embedding the local and global relations of heterogeneous graph for rumor detection," in *ICDM*, 2019, pp. 796–805.

[148] M. Yoon, B. Hooi, K. Shin, and C. Faloutsos, "Fast and accurate anomaly detection in dynamic graphs with a two-pronged approach," in *KDD*, 2019, pp. 647–657.

[149] B. Zong, Q. Song, M. R. Min, W. Cheng, C. Lumezanu, D. Cho, and H. Chen, "Deep autoencoding gaussian mixture model for unsupervised anomaly detection," in *ICLR*, 2018.

[150] G. Pang, C. Shen, and A. van den Hengel, "Deep anomaly detection with deviation networks," in *KDD*, 2019, pp. 353–362.

[151] C. Zhou and R. C. Paffenroth, "Anomaly detection with robust deep autoencoders," in *KDD*, 2017, pp. 665–674.

[152] R. Chalapathy, E. Toth, and S. Chawla, "Group anomaly detection using deep generative models," in *PKDD*, vol. 11051, 2018, pp. 173–189.

[153] Z. Liu, C. Chen, X. Yang, J. Zhou, X. Li, and L. Song, "Heterogeneous graph neural networks for malicious account detection," in *CIKM*, 2018, pp. 2077–2085.

[154] S. Bhatia, B. Hooi, M. Yoon, K. Shin, and C. Faloutsos, "Midas: Microcluster-based detector of anomalies in edge streams," in *AAAI*, 2020, pp. 3242–3249.

[155] S. Dhawan, S. C. R. Gangireddy, S. Kumar, and T. Chakraborty, "Spotting collective behaviour of online frauds in customer reviews," in *IJCAI*, 2019, pp. 245–251.

[156] Q. Huang, C. Zhou, J. Wu, L. Liu, and B. Wang, "Deep spatial–temporal structure learning for rumor detection on twitter," *Neural Computing and Applications*, 08 2020.

[157] L. Ruff, R. A. Vandermeulen, N. Görnitz, A. Binder, E. Müller, K. Müller, and M. Kloft, "Deep semi-supervised anomaly detection," *arXiv preprint arXiv:1906.02694*, 2019.

[158] S. Kim, Y. Tsai, K. Singh, Y. Choi, E. Ibok, C. Li, and M. Cha, "Date: Dual attentive tree-aware embedding for customs fraud detection," in *KDD*, 2020, pp. 2880–2890.

[159] S. Rayana and L. Akoglu, "Less is more: Building selective anomaly ensembles," *ACM Trans. Knowl. Discov. Data*, vol. 10, no. 4, pp. 1–33, 2016.

[160] P. Sen, G. Namata, M. Bilgic, L. Getoor, B. Gallagher, and T. Eliassi-Rad, "Collective classification in network data," *AI Mag.*, vol. 29, no. 3, pp. 93–106, 2008.

[161] P. I. Sánchez, E. Müller, F. Laforet, F. Keller, and K. Böhm, "Statistical selection of congruent subspaces for mining attributed graphs," in *ICDM*, 2013, pp. 647–656.

[162] A. Lancichinetti and S. Fortunato, "Community detection algorithms: a comparative analysis," *Physical review E*, vol. 80, no. 5, p. 056117, 2009.

[163] D. J. Watts and S. H. Strogatz, "Collective dynamics of 'small-world'networks," *nature*, vol. 393, no. 6684, pp. 440–442, 1998.

[164] L. Akoglu and C. Faloutsos, "Rtg: A recursive realistic graph generator using random typing," in *ECML-PKDD*. Springer, 2009, pp. 13–28.

[165] L. Bozarth and C. Budak, "Toward a better performance evaluation framework for fake news classification," in *ICWSM*, 2020, pp. 60–71.

[166] N. Elmrabit, F. Zhou, F. Li, and H. Zhou, "Evaluation of machine learning algorithms for anomaly detection," in *Cyber Security*, 2020, pp. 1–8.

[167] A. H. Engly, A. R. Larsen, and W. Meng, "Evaluation of anomaly-based intrusion detection with combined imbalance correction and feature selection," in *NSS*, vol. 12570, 2020, pp. 277–291.

[168] E. Alsentzer, S. G. Finlayson, M. M. Li, and M. Zitnik, "Subgraph neural networks," in *NIPS*, 2020.

[169] H. Wang, Q. Zhang, J. Wu, S. Pan, and Y. Chen, "Time series feature learning with labeled and unlabeled data," *Pattern Recognition*, vol. 89, pp. 55–66, 2019.

[170] Q. Zhang, J. Wu, P. Zhang, G. Long, and C. Zhang, "Salient subsequence learning for time series clustering," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 9, pp. 2193–2207, 2018.

[171] A. Sankar, Y. Wu, L. Gou, W. Zhang, and H. Yang, "Dysat: Deep neural representation learning on dynamic graphs via self-attention networks," in *WSDM*, 2020, pp. 519–527.

[172] S. M. Kazemi, R. Goel, K. Jain, I. Kobyzev, A. Sethi, P. Forsyth, and P. Poupart, "Representation learning for dynamic graphs: A survey," *J. Mach. Learn. Res.*, vol. 21, pp. 70:1–70:73, 2020.

[173] W. L. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *NIPS*, 2017, pp. 1024–1034.

[174] M. R. Khan and J. E. Blumenstock, "Multi-gcn: Graph convolutional networks for multi-view networks, with applications to global poverty," in *AAAI*, 2019, pp. 606–613.

[175] S. Fan, X. Wang, C. Shi, E. Lu, K. Lin, and B. Wang, "One2multi graph autoencoder for multi-view graph clustering," in *WWW*, 2020, pp. 3070–3076.

[176] J. Cheng, Q. Wang, Z. Tao, D. Xie, and Q. Gao, "Multi-view attribute graph convolution networks for clustering," in *IJCAI*, 2020, pp. 2973–2979.

[177] J. Zhang, B. Cao, S. Xie, C. Lu, P. S. Yu, and A. B. Ragin, "Identifying connectivity patterns for brain diseases via multi-side-view guided deep architectures," in *SDM*, 2016, pp. 36–44.

[178] H. Xiao, J. Gao, D. S. Turaga, L. H. Vu, and A. Biem, "Temporal multi-view inconsistency detection for network traffic analysis," in *WWW*, 2015, pp. 455–465.

[179] S. Bhatia, A. Jain, P. Li, R. Kumar, and B. Hooi, "Mstream: Fast anomaly detection in multi-aspect streams," in *WWW*, 2021, pp. 3371–3382.

[180] E. Gujral, R. Pasricha, and E. Papalexakis, "Beyond rank-1: Discovering rich community structure in multi-aspect graphs," in *WWW*, 2020, pp. 452–462.

[181] N. Shah, A. Beutel, B. Gallagher, and C. Faloutsos, "Spotting suspicious link behavior with fbox: An adversarial perspective," in *ICDM*, 2014, pp. 959–964.

[182] H. Chen, H. Yin, X. Sun, T. Chen, B. Gabrys, and K. Musial, "Multi-level graph convolutional networks for cross-platform anchor link prediction," in *KDD*, 2020, pp. 1503–1511.

[183] A. Guzzo, A. Pugliese, A. Rullo, D. Sacca, and A. Piccolo, "Malevolent activity detection with hypergraph-based models," *IEEE Trans. Knowl. Data Eng.*, vol. 29, no. 5, pp. 1115–1128, 2017.

[184] X. Sun, H. Yin, B. Liu, H. Chen, J. Cao, Y. Shao, and N. Q. Viet Hung, "Heterogeneous hypergraph embedding for graph classification," in *WSDM*, 2021, pp. 725–733.

[185] J. Silva and R. Willett, "Hypergraph-based anomaly detection of high-dimensional co-occurrences," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 3, pp. 563–569, 2008.

[186] X. Wang, J. Xu, T. Zeng, and L. Jing, "Local distribution-based adaptive minority oversampling for imbalanced data classification," *Neurocomputing*, vol. 422, pp. 200–213, 2021.

[187] K. Cheng, S. Gao, W. Dong, X. Yang, Q. Wang, and H. Yu, "Boosting label weighted extreme learning machine for classifying multi-label imbalanced data," *Neurocomputing*, vol. 403, pp. 360–370, 2020.

[188] J. Tang, C. C. Aggarwal, and H. Liu, "Node classification in signed social networks," in *SDM*, 2016, pp. 54–62.

[189] S. Gao, L. Denoyer, and P. Gallinari, "Temporal link prediction by integrating content and structure information," in *CIKM*, 2011, pp. 1169–1174.

[190] V. Sanh, T. Wolf, and S. Ruder, "A hierarchical multi-task approach for learning embeddings from semantic tasks," in *AAAI*, 2019, pp. 6949–6956.

[191] M. Hessel, H. Soyer, L. Espeholt, W. Czarnecki, S. Schmitt, and H. van Hasselt, "Multi-task deep reinforcement learning with popart," in *AAAI*, 2019, pp. 3796–3803.

[192] G. Pang, C. Yan, C. Shen, A. v. d. Hengel, and X. Bai, "Self-trained deep ordinal regression for end-to-end video anomaly detection," in *CVPR*, 2020, pp. 12 173–12 182.

[193] B. Sanchez-Lengeling, J. N. Wei, B. K. Lee, E. Reif, P. Wang, W. W. Qian, K. McCloskey, L. J. Colwell, and A. B. Wiltschko, "Evaluating attribution for graph neural networks," in *NIPS*, 2020.

[194] T. Ide, A. Dhurandhar, J. Navratil, M. Singh, and N. Abe, "Anomaly attribution with likelihood compensation," in *AAAI*, 2021.

[195] F. Hohman, H. Park, C. Robinson, and D. H. P. Chau, "Summit: Scaling deep learning interpretability by visualizing activation and attribution summarizations," *IEEE Trans. Vis. Comput. Graph.*, vol. 26, no. 1, pp. 1096–1106, 2019.

[196] T. Babaie, S. Chawla, S. Ardon, and Y. Yu, "A unified approach to network anomaly detection," in *BigData*, 2014, pp. 650–655.

[197] X. Li, P. Chen, L. Jing, Z. He, and G. Yu, "Swisslog: Robust and unified deep learning based log anomaly detection for diverse faults," in *ISSRE*, 2020, pp. 92–103.

[198] M. W. Mahoney and P. Drineas, "CUR matrix decompositions for improved data analysis," *Proc. Natl. Acad. Sci. USA*, vol. 106, no. 3, pp. 697–702, 2009.

[199] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, “Graph attention networks,” *arXiv preprint arXiv:1710.10903*, 2017.

[200] R. A. Rossi, B. Gallagher, J. Neville, and K. Henderson, “Modeling dynamic behavior in large evolving graphs,” in *WSDM*, 2013, pp. 667–676.

[201] M. Kulldorff, “A spatial scan statistic,” *Commun. Stat.- Theory Methods*, vol. 26, no. 6, pp. 1481–1496, 1997.

[202] D. B. Neill, A. W. Moore, M. Sabhnani, and K. Daniel, “Detection of emerging space-time clusters,” in *KDD*, 2005, pp. 218–227.

[203] J. L. Sharpnack, A. Krishnamurthy, and A. Singh, “Near-optimal anomaly detection in graphs using lovasz extended scan statistic,” in *NIPS*, 2013, pp. 1959–1967.

[204] R. H. Berk and D. H. Jones, “Goodness-of-fit test statistics that dominate the kolmogorov statistics,” *Wahrsch. Verw. Gebiete.*, vol. 47, no. 1, pp. 47–59, 1979.

Appendix A
Challenges in Graph Anomaly Detection

Due to the complexity of anomaly detection and graph data mining, adopting deep learning technologies for graph anomaly detection faces a number of challenges:

Data-CH1. Ground-truth is scarce. In most cases, there is none or little prior knowledge about feature or pattern information of anomalies in real applications. The ground-truth anomalies are often identified by domain experts and this inherently introduces prohibitive cost for assessing the ground truth anomalies. As a result, labeled ground-truth anomalies are often unavailable for analysis in a wide-range of disciplines.

Data-CH2. Various types of graphs. To model real-world data, various types of graphs have been proposed. For instance, graphs can be classified as plain graphs containing only the structural information, and attributed graphs containing both structural information and attribute information. Moreover, to represent the complex relations between different types of objects, heterogeneous graphs are also employed in many applications. For example, online review networks can be represented as heterogeneous graphs to preserve the review relationship between reviewers and items, and contain two types of nodes - reviewer nodes and item nodes [125]. These graphs reflect the real-world data in different forms and graph anomalies will show different deviating patterns in different types of graphs.

Data-CH3. Various types of graph anomalies. Given a specific type of graph, graph anomalies could appear as a specific node, edge, sub-graph, or an entire graph and each type of these anomalies is significantly different from others. This requires the detection methods to provide concise definitions of anomalies and identify concrete clues about the deviating patterns of anomalies.

Data-CH4. High dimensionality and large scale. Representing the structure information of real-world networks usually results in high dimensional and large-scale data [35] because real-world network often contain millions or billions of nodes, such as Facebook and Twitter. The graph anomaly detection techniques, hence, should be capable of handling such high dimensional and large scale data, and extract the patterns of anomalies under the constraints of execution time and current computing resources.

Data-CH5. Interdependencies and dynamics. The relationships between real objects reveal their interdependencies and they can no longer be treated individually for anomaly detection. That is to say, the detection techniques need to digest the deviating patterns of anomalies by considering the pairwise, triadic, and higher relationships among objects restored in conventional graphs or hypergraphs [182]–[185]. In addition, the dynamic nature of real-networks makes the detection problem much more challenging when the graph structure and attributes of nodes or edges evolve overtime.

Data-CH6. Class imbalance. As anomalies are rare occurrences in real-world, only a very small proportion of the whole data might be anomalous. This naturally introduces a critical class imbalance problem to anomaly detection because the number of anomalies is far smaller than normal objects in the training data. If no further actions are taken to tackle this challenge, the learning-based anomaly detection techniques might overlook the patterns of anomalies and lead to sub-optimal results.

Data-CH7. Unknown and camouflage of anomalies. In reality, the knowledge about anomalies mainly stems from our expertise experiences. There are still many unknown anomalies across different application domains and new types of anomalies might appear in the future. Nevertheless, real-world anomalies also appear to hide or camouflage them as benign objects to bypass existing detection systems. In graphs, anomalies might hide themselves by connecting with many normal nodes or mimic their attributes. The detection methods, hence, should be adaptive to unknown and novel anomalies and robust to camouflaged anomalies.

TABLE 7: Challenges and Methods.

Challenges	Details	Methods
Data-specific Challenges		
Data-CH1	Ground-truth is scarce	[41], [81]–[83], [91], [96], [98], [101], [105], [106], [111], [118], [122], [132]
Data-CH2	Various types of graphs	[27], [81], [82], [90], [91], [95], [98], [101], [102], [104], [107], [108], [125], [127], [134]
Data-CH3	Various types of graph anomalies	[27], [91], [95], [98], [119]
Data-CH4	High dimensionality and large scale	[76], [81], [83], [101], [103], [104], [132], [149]
Data-CH5	Interdependencies and dynamics	[76], [81]–[83], [90], [91], [95], [96], [98], [101]–[105], [110], [118], [119], [122], [125], [127], [132], [134], [142]
Data-CH6	Class imbalance	[66], [105], [135], [149]
Data-CH7	Unknown and camouflage of anomalies	[26], [27], [41], [101], [102], [106], [122], [125], [127], [132]
Techniques-specific Challenges		
Tech-CH1	Anomaly-aware training objectives	[27], [66], [72], [76], [81]–[83], [90], [91], [95], [96], [98], [101]–[108], [110], [111], [118], [119], [122], [125], [127], [132], [134], [135], [142]
Tech-CH2	Anomaly interpretability	[76], [105]
Tech-CH3	High training cost	[72], [83], [91], [96], [117], [125]
Tech-CH4	Hyperparameter tuning	[66], [69]

We summarize these data-specific challenges and technical-specific challenges (discussed in Section 1.1) as well as corresponding works that aim to address them in Table. 7.

Appendix B
Taxonomy

The taxonomy of this survey is shown in Fig. 10.

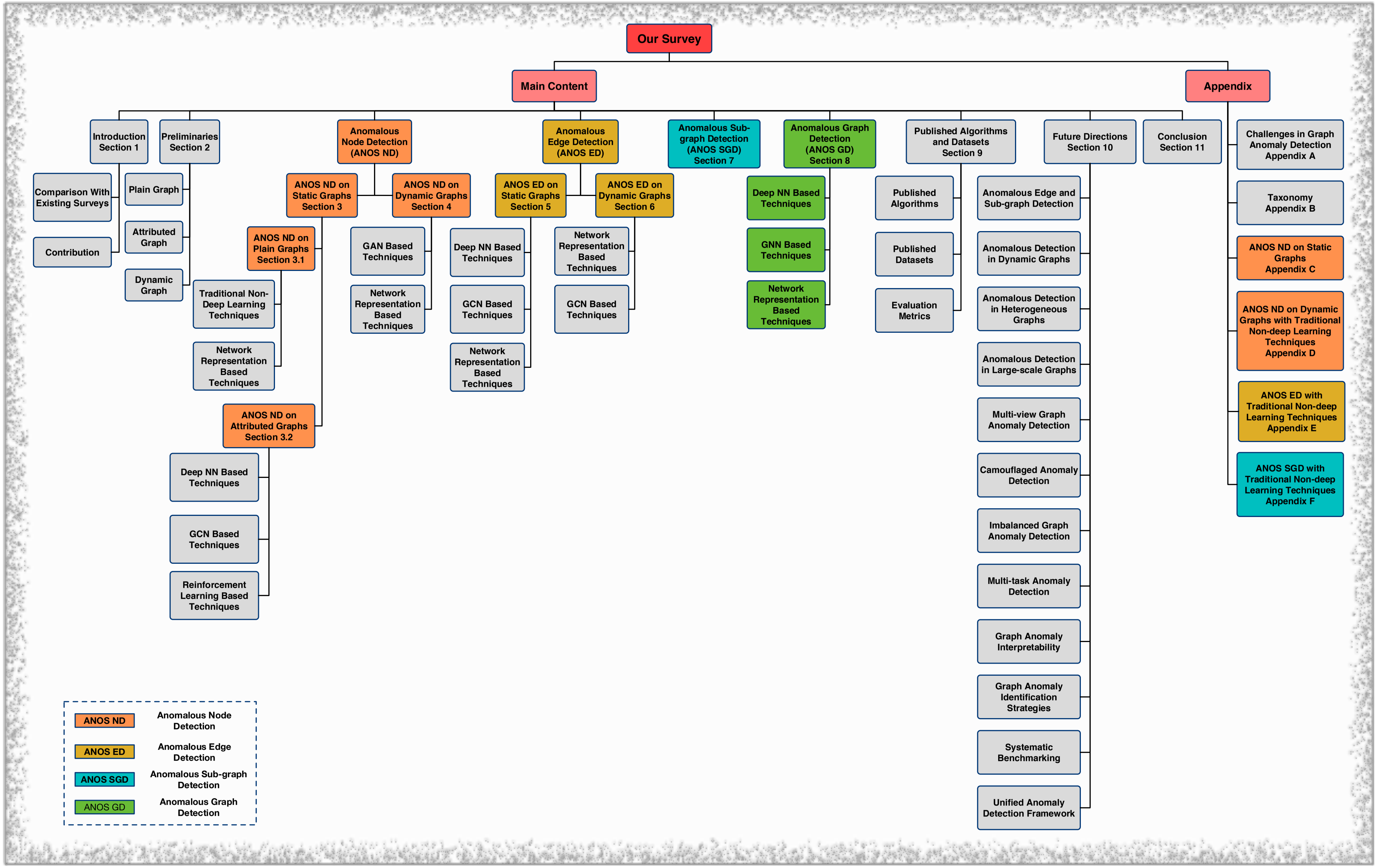

Our Survey
Main Content
Appendix
Introduction Section 1
Comparison With Existing Surveys
Contribution
Preliminaries Section 2
Plain Graph
Attributed Graph
Dynamic Graph
Anomalous Node Detection (ANOS ND)
ANOS ND on Static Graphs Section 3
ANOS ND on Plain Graphs Section 3.1
Traditional Non-Deep Learning Techniques
Network Representation Based Techniques
ANOS ND on Attributed Graphs Section 3.2
Deep NN Based Techniques
GCN Based Techniques
Reinforcement Learning Based Techniques
ANOS ND on Dynamic Graphs Section 4
GAN Based Techniques
Network Representation Based Techniques
Anomalous Edge Detection (ANOS ED)
ANOS ED on Static Graphs Section 5
Deep NN Based Techniques
GCN Based Techniques
Network Representation Based Techniques
ANOS ED on Dynamic Graphs Section 6
Network Representation Based Techniques
GCN Based Techniques
Anomalous Sub-graph Detection (ANOS SGD) Section 7
Anomalous Graph Detection (ANOS GD) Section 8
Deep NN Based Techniques
GNN Based Techniques
Network Representation Based Techniques
Published Algorithms and Datasets Section 9
Published Algorithms
Published Datasets
Evaluation Metrics
Future Directions Section 10
Anomalous Edge and Sub-graph Detection
Anomalous Detection in Dynamic Graphs
Anomalous Detection in Heterogeneous Graphs
Anomalous Detection in Large-scale Graphs
Multi-view Graph Anomaly Detection
Camouflaged Anomaly Detection
Imbalanced Graph Anomaly Detection
Multi-task Anomaly Detection
Graph Anomaly Interpretability
Graph Anomaly Identification Strategies
Systematic Benchmarking
Unified Anomaly Detection Framework
Conclusion Section 11
Challenges in Graph Anomaly Detection Appendix A
Taxonomy Appendix B
ANOS ND on Static Graphs Appendix C
ANOS ND on Dynamic Graphs with Traditional Non-deep Learning Techniques Appendix D
ANOS ED with Traditional Non-deep Learning Techniques Appendix E
ANOS SGD with Traditional Non-deep Learning Techniques Appendix F
ANOS ND Anomalous Node Detection
ANOS ED Anomalous Edge Detection
ANOS SGD Anomalous Sub-graph Detection
ANOS GD Anomalous Graph Detection

Fig. 10: The Taxonomy of Our Survey.

APPENDIX C
ANOS ND ON STATIC GRAPHS

Following the taxonomy of Section 3.2, we review traditional non-deep learning techniques, GAT (graph attention) based techniques, GAN (generative adversarial network) based techniques, and network representation based techniques that devote to ANOS ND on static attributed graphs as follows.

C.1 Traditional Non-Deep Learning Techniques

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, and achieved promising detection performance.

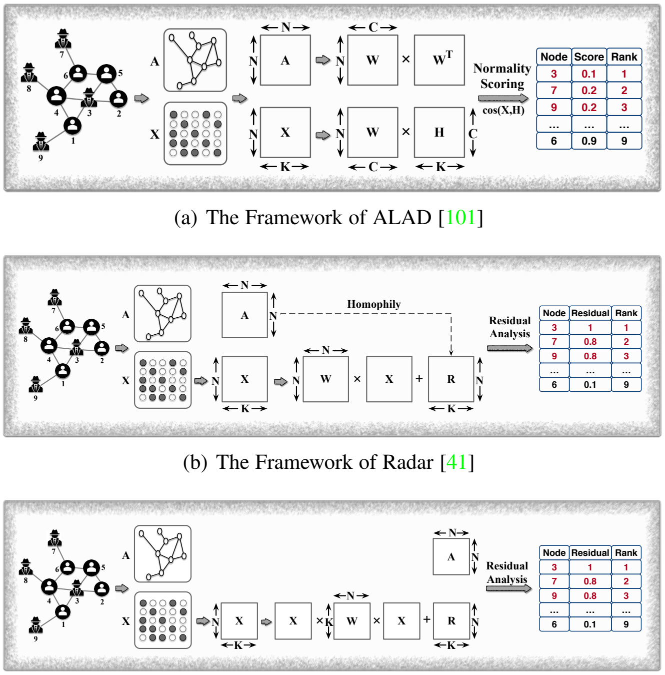
An early attempt is Liu et al. [101]. They aimed to detect community anomalies (prior defined in Section 3) through the developed model, ALAD. As shown in Fig. 11(a), ALAD incorporates both the graph structure A and node attributes X to derive the community structures W and their attribute distribution vector H through non-negative matrix factorization. When the matrices are decomposed, ALAD measures the normality of each node according to attribute similarity between it and its belonging community. By ranking the nodes' normality scores in ascending order, the top-k nodes are identified as community anomalies.

Li et al. [41] approached ANOS ND from a different perspective by using residual analysis. As assumed, anomalies will lead to larger attribute reconstruction residual errors because they do not conform to the attribute patterns of the majorities. Accordingly, residual errors R, as shown in Fig. 11(b), are learned through factorizing the node attributes X by the proposed model, Radar. To incorporate the structural information for obtaining these errors, Radar puts explicit restriction on the learned residuals such that directly linked nodes will have similar residual patterns in R (known as homophily effect). Finally, the top-k nodes with larger norms in R are identified as anomalies.

Although the homophily hypothesis provides strong support to exploit the structural information, it might not always be held. In fact, real objects would experience distinctive attributes to their connected neighbors and it is non-trivial to regulate all connected objects share similar values in each dimension in the feature space. By this, Peng et al. [102] indicated that there are structurally irrelevant node attributes that does not satisfy the homophily hypothesis. Indeed, these structurally irrelevant node attributes would have adverse effects on anomaly detection techniques that are developed based on this hypothesis. To tackle this problem, their developed model, ANOMALOUS, utilizes CUR [198] decomposition to select attributes that are closely related to network structure and then spot anomalies through residual analysis following Radar (as shown in Fig. 11(c)).

Apart from MF, linear regression models are also adopted to train anomaly classifiers given the labeled training data. A representative work is Wu et al. [103]. Their supervised model, SGASD, has yield encouraging results on identifying social spammers utilizing the social network structure, content information in social media and user labels.

These non-deep learning techniques could capture valuable information from the graph topology and node attributes, but their applications and generalizabilities to real networks (usually in large-scale) are strictly limited due to the high computational cost of the matrix decomposition operation and regression models.



(a) The Framework of ALAD [101]

(b) The Framework of Radar [41]

(c) The Framework of ANOMALOUS [102]

Fig. 11: ANOS ND on attributed graphs – Matrix Factorization based approaches. The three representative MF techniques adopt different decomposition strategies to extract valuable information from the graph structure and node attributes. Anomalous nodes are then spotted through scoring functions or residual analysis.

C.2 GAT Based Techniques

Although GCN provides an effective solution to incorporate the graph structure with node attributes for ANOS ND (reviewed in Section 3.2.2), due to the simple convolution operation that aggregates neighbor information equally to the target node, GCN's capability in capturing the most relevant information from neighbors is unsatisfactory. In more recent works, the graph attention mechanism (GAT) [199] is employed to replace the traditional graph convolution.

For instance, Fan et al. [104] applied graph attention neural network to encode the network structure information (structure encoding). The method, AnomalyDAE, also adopts a separate attribute autoencoder to embed node attributes (attribute encoding). Through the unsupervised encoding-decoding process, each node is ranked according to its corresponding reconstruction loss, and the top-k nodes introducing greatest losses are identified as anomalies. Specifically, as shown in Fig. 12, the attribute decoding process takes both node embeddings learned through the structure and attribute encoding processes to reconstruct node attributes, while the graph topology is reconstructed only using the embeddings output by the GAT. To acquire better reconstruction results, AnomalyDAE is trained to minimize the overall loss function which can be denoted as:

$$\begin{aligned}\mathcal{L}_{AnomalyDAE} =& \alpha||(A-\hat{A})\odot\boldsymbol{\theta}||_2^2 + \\ & (1-\alpha)||(X-\hat{X})\odot\boldsymbol{\eta}||_2^2,\end{aligned} \tag{19}$$

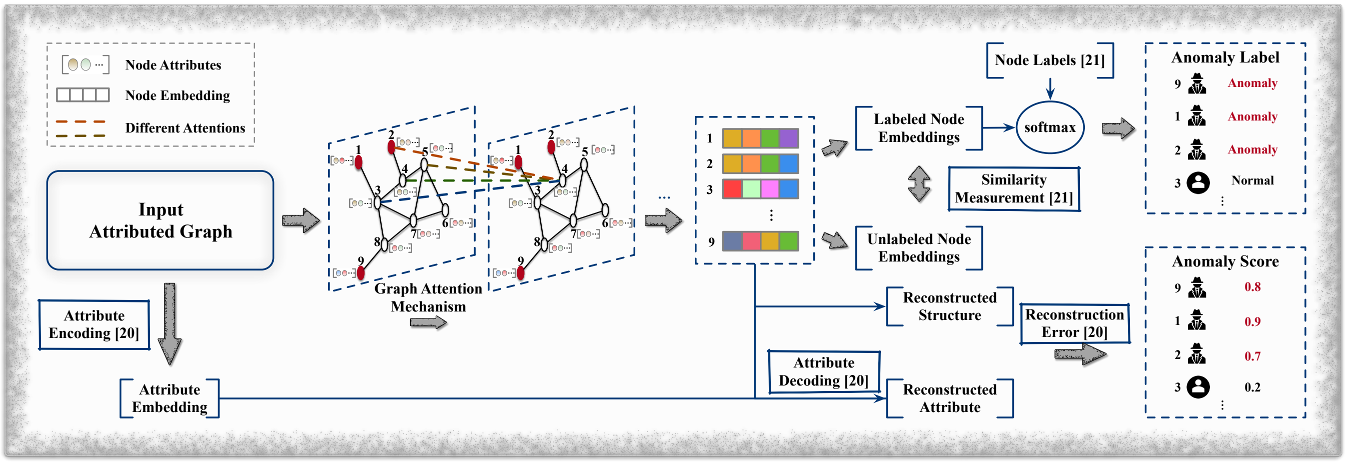


Fig. 12: ANOS ND on attributed graphs – GAT based approaches. Given the input graph, these techniques employ graph attention neural network to learn node embeddings. The unsupervised techniques, AnomalyDAE [104] scores each node based on the reconstruction loss and mark the top-k nodes as anomalies, while the semi-supervised techniques, SemiGNN [105] trains a classifier to predict node labels.

where α is the coefficient, A and X is the input adjacency matrix and attribute matrix, $\hat{A}$ and $\hat{X}$ are the reconstructed matrices. Each $\theta_{i,j} \in \boldsymbol{\theta}$ and $\eta_{i,j} \in \boldsymbol{\eta}$ is 1, if the corresponding element A_{ij} and X_{ij} equals 0, otherwise, their values are defined by hyperparameters greater than 1.

Another decent work is SemiGNN [105], in which Wang et al. proposed a semi-supervised attention based graph neural network to detect fraudulent users in online payment platforms. This work further explores user information collected from various sources (e.g., transaction information and user profiles), and represents the real-networks as multi-view graphs. Each view in the graph is modeled to reflect the relationship between users or the correlation between user attributes. For anomaly detection, SemiGNN first generates node embedding h_u^v from each view v by aggregating neighbors' information through a node-level attention mechanism. It then employs a view-level attention to aggregate node embeddings from each view, and generates an unified representation a_u for each node. Lastly, the class of each node is predicted through a softmax classifier. Indeed, Wang et al. designed a supervised classification loss and an unsupervised graph reconstruction loss to jointly optimize the model by fully utilizing labeled and unlabeled data. The classification loss can be denoted as:

$$\mathcal{L}_{sup} = -\frac{1}{|U_L|} \sum_{u \in U_L} \sum_{i=1}^{k} I(y_u = i) \log \frac{\exp(a_u \cdot \theta_i)}{\sum_{j=1}^{k} \exp(a_u \cdot \theta_j)}, \tag{20}$$

where U_L is the labeled user set and its size is $|U_L|$, $I(\cdot)$ is an indicator function, k is the number of labels to be predicted (in most cases, the label is either anomalies or non-anomalies, and $k = 2$) and θ are trainable variables. Meanwhile, the unsupervised loss is proposed to encourage unlabeled nodes (users) that can be reached by labeled nodes through random walks to obtain similar representations, and vice versa. This is achieved by negative sampling (unlabeled nodes that cannot be reached by random walks are negative samples) and the loss can be formulated as:

$$\begin{aligned} \mathcal{L}_{unsup} = \sum_{u \in U} \sum_{v \in N_u \cup Neg_u} & -\log(\sigma(a_u^T a_v)) \\ & - 3 \cdot E_{q \sim P_{neg}(u)} \log(\sigma(a_u^T a_q)), \end{aligned} \tag{21}$$

where U is the user set, N_u is the neighbor set of u, Neg_u are negative samples, P is the sampling distribution, and $\sigma(\cdot)$ is the sigmoid function. The total loss takes the sum of them and can be denoted as:

$$\mathcal{L}_{SemiGNN} = \alpha \mathcal{L}_{sup} + (1 - \alpha) \mathcal{L}_{unsup} + \lambda \mathcal{L}_{reg}, \tag{22}$$

where α is a balancing parameter and $\mathcal{L}_{reg}$ regularizes all trainable variables.

C.3 GAN Based Techniques

For the effectiveness of GAN in capturing anomalous/regular data distributions (as reviewed in Section 4.2), Ding et al. [106] applied GAN in their developed model, AEGIS, for improved anomaly discriminability on unseen data. As shown in Fig. 13, this model first generates node embeddings through GNN from the input attributed graph and then trains a generator and discriminator which can identify anomalies. The first phase is expected to map anomalous nodes and regular nodes to distinctive areas in the embedding space such that the GAN can learn a boundary between them. Accordingly, Ding et al. built an autoencoder network with graph differentiative layers which can capture the attribute difference between each node and its K^{th}-order neighbors. Such difference information enables anomalies to be easier distinguished and is encoded into the embeddings following:

$$h_i^l = \sum_{k=1}^{K} \beta_i^k \sigma \left(W_1 h_i^{l-1} + \sum_{j \in N_k(i)} \alpha_{ij} W_2 \Delta_{ij}^{l-1} \right), \tag{23}$$

where h_i^l is the learned representation of node i by the l-th layer, β_i^k is the attention for each hop, $N_k(i)$ is the set of k-th order neighbors, α_{ij} is the attention for each neighbor and Δ_{ij}^{l-1} is the difference between i and j, W_1 and W_2 are trainable variables. This autoencoder is fine-tuned until the node attributes can be best reconstructed using the learned embeddings, after which the GAN is trained.

In the second phase, the generator tries to generate anomalies by sampling noisy data from a prior distribution $p(\tilde{z})$, while the discriminator struggles to distinguish normal nodes' embeddings and the generated anomalies. The training process of GAN can

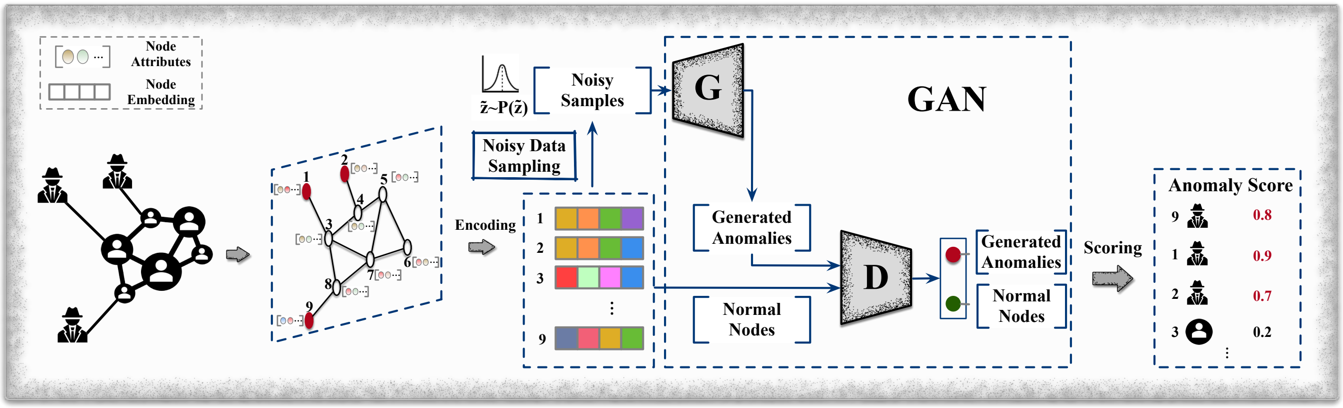


Fig. 13: ANOS ND on static graphs – GAN based approaches. The generator, G, generates potential anomalies by sampling noisy data from a prior distribution to fool the discriminator. The discriminator, D, distinguishes the generated anomalies and normal nodes. The scoring function then quantifies the anomaly score of each node according to D's output.

be formulated as the mini-max game between the generator and discriminator as follows:

$$\min_{G}\max_{D}\mathbb{E}_{z\sim Z}[\log(D(z))]+\mathbb{E}_{\tilde{z}\sim\tilde{Z}}[\log(1-D(G(\tilde{z})))], \quad (24)$$

where z is the node embeddings, $\tilde{z}$ is the generated anomalies. After training, AEGIS directly learns embedding z_u for a test node u, and quantifies its anomaly score with regard to the discriminator's outputs - the possibility of the node to be normal. The scoring function hence is formulated as:

$$Ascore(u) = 1 - D(z_u), \quad (25)$$

and the top-k nodes are anomalous.

C.4 Network Representation Based Techniques

Following the routine of encoding graphs into an vector space, after which anomalies are detected (reviewed in Section 3.1.2), plenty of studies on ANOS ND in attributed graphs have exploited deep network representation techniques.

For instance, Zhang et al. [107] detected abnormal nodes that have attributes significantly deviating from their neighbors through a 3-layer neural network, REMAD, and residual analysis. Given the original node attribute matrix, they explicitly divide it into a residual attribute matrix R and a structurally relevant attribute matrix $\hat{X}$ that are proposed to capture the abnormal characters of anomalies and for network representation learning, respectively. Both matrices are jointly updated throughout the representation learning process to encourage nearby nodes have similar representations. Specifically, these node embeddings are generated by aggregating neighbors' information with each node's own attributes and this can be denoted as:

$$h_i^l = \sigma\left(W^l \cdot \text{CONCAT}\{h_i^{l-1}, h_{N_i}^l\} + b^2\right), \quad (26)$$

where h_i^l is node i's representation generated by the l-th layer ($h_i^0 = \hat{X}$), N_i contains i's neighbors, $\sigma()$ is the activation function, W^l and b are trainable variables. Finally, the residual matrix R will contain the abnormal information of each node and the top-k nodes with largest norms are anomalies.

Given part of node labels, Liang et al. [108] developed a semi-supervised representation model, SEANO, to incorporate graph structure, node attributes and label information. Similar to REMAD, SEANO also aggregates neighbor information to center nodes, and the node representations are obtained through an embedding layer, which can be denoted as:

$$z_i = \lambda_i h^{l_1}(x_i) + (1-\lambda_i)h^{l_1}(\bar{x}_{N_i}), \quad (27)$$

where z_i is i's representation, λ_i is a trainable variable that identifies the weight of i's own attributes (x_i), $\bar{x}_{N_i}$ is the average of node i's neighbors' representations, and the function $h^k(x_i) = \phi(W^k h^{k-1}(x_i) + b^k)$ maps original node attributes into lower dimensional vectors. Then, a supervised component which takes the representations as input predicts node labels through a softmax classifier, and an unsupervised component is trained to reconstruct node contexts (node sequences). For each node, its context is generated not only through random walks on the graph but also contains labeled nodes that belong to the same class, if it is labeled. After training, SEANO interprets λ_i as the normality score of node i and the top-k nodes with highest scores are anomalies.

Learning node representations via aggregating neighbor information is effective in capturing comprehensive information from graph structure and node attributes for ANOS ND. But, Liu et al. [144] demonstrated such an approach would help anomalies aggregate features from regular nodes, making them look normal and leading to sub-optimal detection performance. They identified three concrete issues that should be accounted when applying the aggregation operation for anomaly detection: 1) Anomalies are rare objects in the network, directly aggregating neighborhoods' information would smooth the difference between anomalies and normal instances, and blurs the boundaries between them. 2) Directly connected nodes would have distinctive features and the assumption that connected nodes share similar features, serving as the basis for feature aggregation, no longer holds in this scenario. 3) Real objects also form multiple types of relations with others, aggregation results are distinctive following different relations. With regard to these concerns, their proposed method GraphConsis takes a sampling strategy to avoid potential anomalous neighbors when aggregating node features. This method also adopts an attention mechanism to aggregate neighbor information following different links. The learned node representations, therefore, are more robust to anomalies and GraphConsis takes them as input to train a classifier for predicting labels.

Dou et al. [27] further considered camouflage behaviors of fraudsters in their proposed model CARE-GNN to enhance the detection performance. As specified, the camouflages can be categorized as feature camouflage and relation camouflage that anomalies adjust their feature information or form connections with many benign objects to gloss over suspicious information, respectively. Hence, directly employing aggregation will overlook the camouflages and smooth the abnormal patterns of anomalies, eliminating the distinctions between anomalies and normal objects. To alleviate such over-smoothness, CARE-GNN also adopts a neighbor sampling strategy as GraphConsis to filter camouflaged anomalies and explores different types of relations formed between users. Specifically, under each relation, Dou et al. employed a MLP to predict node labels using their features and measure the similarity ($l1$ distance) between each node and its neighbors according to the MLP's output. Then, the top-k most similar neighbors are selected for feature aggregation and CARE-GNN generates each node's representation through a combination of latent representations that are learned under different relations and a classifier is eventually trained using the representations to predict node labels.

As can be seen, these network representation based techniques' performance are decided by their training objectives/loss functions. An enhanced detection performance can be foreseen if the loss function could well-separate normal/abnormal nodes. Motivated by this, a more recent work in [66] emphasizes the importance of anomaly-aware loss functions. In order to adjust margins for the anomalies, the authors proposed a novel loss function to guide the representation learning process. Specifically, this loss function is designed to find the relative scales between the margins of outlier nodes and normal nodes. A MLP-based classifier is finally trained using the node representations generated by the anomaly-aware loss guided GNNs and node labels. For unseen nodes, the classifier will label them upon their representations.

Appendix D
ANOS ND on Dynamic Graphs With Traditional Non-Deep Learning Techniques

In order to detect anomalous nodes in dynamic plain graphs, traditional non-deep learning techniques rely heavily on modeling nodes' structural evolving patterns. Representative works like [200] and [112] assume that regular nodes' evolutionary behaviors (i.e., generate or remove connections with others) usually follow stable patterns and their changes introduce less impact on the graph structure compared to anomalies. Specifically, in [112], Wang et al. proposed a novel link prediction method to fit the evolutionary patterns of most nodes such that anomalies can be identified because their observations confront significantly with the prediction results. They further quantified the impact of anomalies' behaviors by assessing the perturbation they have posed to the graph adjacency matrix.

Other traditional works also exploit node/edge attributes and their changes. For examples, Teng et al. [110] took node and edge attributes as two different views to describe each node. By encoding both information into a shared latent space, their proposed model learns a hypersphere from historical records. When new observations of existing nodes are given, the model could distinguish their anomalousness according to their distances to the hypersphere centroid and these lying outside the hypersphere are spotted as anomalies. Different from the embedding techniques, in [5], Nguyen et al. proposed an non-parametric method to detect anomalous users, tweets, hashtags and links on social platforms.

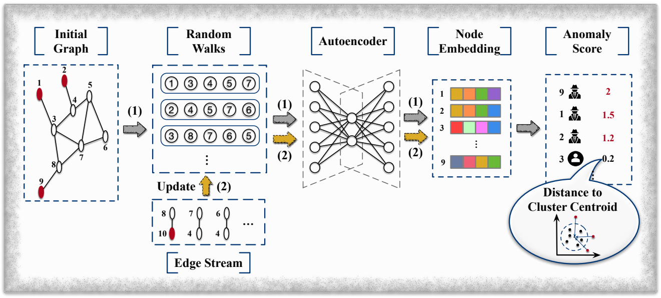


Fig. 14: The Framework of NetWalk [83]. The input dynamic graph is modeled as an initial graph with an incoming edge stream. Given the initial graph, a trunk of node sequences is generated using random walks on the graph and the deep autoencoder encodes each node into an embedding space following process (1). The node sequences and node embeddings are updated based on the incoming edge stream following process (2). Finally, NetWalk assigns an anomaly score to each node according to its distance to the cluster centroid in the embedding space.

Specifically, they modeled social platforms as heterogeneous social graphs such that the affluent relationships between users, tweets, hashtags and links can be effectively captured. Through extensive analysis on the features, such as users' registration information, keywords in tweets, linguistic style of links and popularity scores of hashtags, anomalous objects are spotted based on their deviating features. This work also utilizes relationships between individual object as well as the detected anomalies, and detects groups of abnormal objects.

Appendix E
ANOS ED With Traditional Non-Deep Learning Techniques

Traditional non-deep learning based approaches mainly focus on utilizing temporal signals (e.g., changes in graph structure), and applying specially designed statistical metrics to detect anomalous edges on dynamic graphs [33], [121]. As a concrete example, Eswaran and Faloutsos [141] modeled a dynamic graph as a stream of edges and exploit the graph structure as well as the structure evolving patterns. They identify two signs of anomalous edges: 1) connecting regions of the graph that are disconnected, and 2) appearing in bursts. For incoming edges, their model assigns anomaly scores to each edge and the top-k edges with highest scores are anomalies. Another most recent work by Chang et al. [117] proposed a novel frequency factorization algorithm, aiming to spot anomalous incoming edges based on their likelihood of observed frequency. Specifically, this method merges the advantage of probabilistic models and matrix factorization for capturing both temporal and structural changes of nodes, and as reported, it only requires constant memory to handle edge streams.

Appendix F
ANOS SGD With Traditional Non-Deep Learning Techniques

F.1 ANOS SGD on Static Graphs

One motivation of ANOS SGD in static graphs is that anomalous sub-graphs often exhibit significantly different attribute distributions. These traditional non-deep learning techniques, such as gAnomaly [40], AMEN [12], and SLICENDICE [143], focus on modeling the attribute distributions and measure the normality of sub-graphs. Another line is graph residual analysis. The affluent attribute information contained in real-world networks provides insight for the relationship formed between objects and this motivates several studies to spot anomalous sub-graphs via measuring the residual between the expected structure and observed structure [10].

F.2 ANOS SGD on Dynamic Graphs

For ANOS SGD, traditional works have put great efforts to extract metrics for anomalous sub-graph detection. For instance, [36] introduces six metrics to identify community-based anomalies, namely, grown community, shrunken community, merged community, split community, born community and vanished community. Although these hand-crafted features or statistical patterns well-fit some particular types of existing anomalies, their capability to detect unseen and camouflage anomalies are strongly restricted and applying them directly might introduce high false negative rate which is crucial to applications like financial security. Other works, such as SPOTLIGHT [39] and [30], explore sudden changes in dynamic graphs and identify anomalous sub-graphs that are related to such changes.

Motivated by the phenomena that social spam and fraud groups often form temporal dense sub-graphs in online social networks, plenty of works, including, [9], [29], utilize manual extracted features and spot anomalous dense sub-graphs that have evolve significantly different from the reset part of the graph.

Apart from these, a large number of previous studies attempt to utilize various graph scan statistics for anomalous sub-graph detection, such as the Kulldorff statistic [201], Poisson statistic [202], elevated mean scan statistic [203] and Berk-Jones statistic [204]. Specifically, Shao et al. [124] proposed a non-parametric method to detect anomalous sub-graphs in dynamic graphs where the network structure is constant but node attributes are changing overtime. This works measures the anomalous score of each sub-graph with regard to the p-values of nodes that are comprised in it and sub-graphs with higher scores are more likely to be anomalous. Another work, GBGP [37], instead, adopts the elevated mean scan statistic to identify nodes that might form anomalous sub-graphs and detects anomalous groups that follow predefined irregular structures.