Trustworthy Graph Neural Networks: Aspects, Methods and Trends

He Zhang, Bang Wu, Xingliang Yuan, Shirui Pan, Hanghang Tong, *Fellow, IEEE*, Jian Pei, *Fellow, IEEE*

***Abstract*—Graph neural networks (GNNs) have emerged as a series of competent graph learning methods for diverse real-world scenarios, ranging from daily applications like recommendation systems and question answering to cutting-edge technologies such as drug discovery in life sciences and n-body simulation in astrophysics. However, task performance is not the only requirement for GNNs. Performance-oriented GNNs have exhibited potential adverse effects like vulnerability to adversarial attacks, unexplainable discrimination against disadvantaged groups, or excessive resource consumption in edge computing environments. To avoid these unintentional harms, it is necessary to build competent GNNs characterised by trustworthiness. To this end, we propose a comprehensive roadmap to build trustworthy GNNs from the view of the various computing technologies involved. In this survey, we introduce basic concepts and comprehensively summarise existing efforts for trustworthy GNNs from six aspects, including robustness, explainability, privacy, fairness, accountability, and environmental well-being. Additionally, we highlight the intricate cross-aspect relations between the above six aspects of trustworthy GNNs. Finally, we present a thorough overview of trending directions for facilitating the research and industrialisation of trustworthy GNNs.**

***Index Terms*—Graph neural networks, trustworthy machine learning, robustness, explainability, privacy, fairness, accountability, environmental well-being.**

I. Introduction

GRAPHS are a ubiquitous data structure used to represent data from a broad spectrum of real-world applications. They have demonstrated a remarkable capacity to represent both objects and the diverse interactions between them. For example, a single graph depicting friendship between users on Facebook can have 2.9 billion nodes and more than 490 billion edges [1]. Recently, graph neural networks (GNNs) [2], [3] have emerged as a series of powerful machine learning methods for exploring graph data, and have made impressive advancements in areas ranging from daily applications to technological frontiers. In consumer applications, GNNs enrich personalised search and recommendations for customers on e-commerce (e.g., Alibaba [4]) and social media (e.g., Pinterest [5]) platforms by considering user-user/item interactions. In the advanced sciences, researchers use graphs to represent complex systems (e.g., physics simulations [6]), and employ GNNs to explore the objective laws that govern celestial motion [7]. GNNs are also conducive to improving well-being in our society, with applications ranging from fake news detection [8] to discovering drugs that treat COVID-19 [9].

Many architectures and methods have been proposed to improve the performance of GNNs from different perspectives, including enhancing their expressive power [16], overcoming over-smoothing issues [17], increasing architecture depths [18], and utilising self-supervised signals [19], to name only a few. In critical and sensitive domains, however, competent performance is not the only objective. For example, in the context of GNN-based anomaly detection systems, it is crucial for such systems to be robust against adversarial attacks [20]; in GNN-based credit scoring systems, it can be unethical and raise societal concerns if such systems decline a loan application based on potential biases defined by attributes such as gender, race, and age [21]; in GNN-based drug discovery systems, it is desirable for the process of identifying screened drug candidates to be explainable and comprehensible by humans [22]. Driven by these diverse requirements, people increasingly expect GNNs to be reliable, responsible, ethical, and socially beneficial enough to be trusted. In this survey, we aim to provide a timely survey to summarise these efforts from the perspective of "trustworthy GNNs".

A. Trustworthiness

The core of a trustworthy system lies in its trustworthiness. As defined in the Oxford Dictionary, "trustworthy" describes an object or a person "that you can rely on to be good, honest, sincere, etc." [23]. Synonyms of trustworthy include trustable, reliable, dependable, faithful, honourable, creditworthy, responsible, etc. The trustworthiness of a system essentially builds on the "trust" that makes the system "worthy" of being relied on. There are a number of fields in which "trust" is defined and studied, including philosophy, psychology, sociology, economics, technology, and organisational management [12], [10], [13]. It is widely recognised that trust relates to people's perceived ability to rely on others (e.g., persons or systems); trust, in fact, can be considered the foundation of social order [24].

Trust can also be defined as "the relationship between a trustor and a trustee: the trustor trusts the trustee" [12]. In the context of trustworthy GNNs, we define the trustee as the GNN models or systems and the trustor as the model owners, users, regulators, or even our society as a whole. A GNN model (system) is trustworthy if it is developed with the key aspects of trustworthiness in mind, as we will discuss later.

H. Zhang, B. Wu, X. Yuan, S. Pan are with the Faculty of IT, Monash University, Clayton, VIC 3800, Australia. Emails: {he.zhang1, bang.wu, xingliang.yuan, shirui.pan}@monash.edu. *(Corresponding Author: Shirui Pan).*

H. Tong is with the Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, IL 61801, US. Email: htong@illinois.edu.

J. Pei is with the School of Computing Science, Simon Fraser University, Canada. Email: jpei@cs.sfu.ca.

TABLE I
THE VIEW AND CONTENTS OF TYPICAL CONCEPTS RELATED TO TRUSTWORTHY AI.

View	REF	Robustness	Explainability	Privacy	Fairness	Accountability	Well-being	Others
Technology	[10] [11] [12]	Safety, Robustness, Adversarial Robustness	Interpretability, Explainability, Transparency	Privacy, Consent, Privacy Protection	Non-discrimination, Fairness	Accountability, Auditability, Reproducibility, Transparency	Environmental Well-being, Value Alignment, Professional Codes & Ethics Guidelines, Lived Experience, Disinformation & Filter Bubbles, Social Good	Generalisation, Distribution Shift
Mechanism	[13]	Red Teaming Exercises	Interpretability	PPML Standardization	–	Bias Safety Bounties, Sharing of AI Incidents, Audit Trail	High-Precision Compute Measurement	–
Guideline	[14]	Harmony & Human-Friendly, Fairness & Justice, Inclusion & Sharing, Respect for Privacy, Safety & Controllability, Shared Responsibility, Open & Collaboration, Agile Governance						
	[15]	Well-Being, Respect for Autonomy, Privacy & Intimacy, Solidarity, Democratic Participation, Equity, Diversity Inclusion, Prudence, Responsibility & Sustainable Development						

B. From Trustworthy AI to Trustworthy GNNs

In recent years, a global consensus has developed as to the importance of building trustworthy AI. Continuing efforts have been made to promote the development of ethical, beneficial, responsible, and trustworthy AI [10]. These efforts are presented from views ranging from guidelines and mechanisms to technology (as shown in Table I). In Table I, we present the views and contents of existing efforts related to trustworthy AI.

The research focusing on *guidelines* proposes a set of principles and requirements for building responsible AI [14], [15]. For example, the safety and controllability principle [14] indicates that "the transparency, interpretability, reliability, and controllability of AI systems should be improved continuously to make the systems more traceable, trustworthy, and easier to audit and monitor". From the view of developing *mechanisms*, Brundage *et al.* [13] made recommendations for supporting verifiable claims. For example, no formal process or incentive exists for individuals who are not affiliated with a particular AI developer to report safety- and bias-related issues in AI systems; therefore it is relatively rare for AI systems to be broadly scrutinised on these issues. To this end, the authors recommended an institutional mechanism in which developers of trustworthy AI systems should pilot bias and safety bounties. From the *technology* perspective, a few studies have summarised advancements in trustworthiness derived from principles outlined by existing guidelines [10], [11]. For example, the privacy of trustworthy AI can be enhanced by confidential computing [25], federated learning [26], and differential privacy technologies [27], [10].

Although the existing literature explores the space of trustworthy AI from different views (i.e., "Technology", "Mechanism", "Guideline" in Table I), several key aspects from the technology view receive the most recognition, namely, robustness, explainability, privacy, fairness, accountability, and well-being. As a vital instantiation of trustworthy AI in the context of GNNs, we transfer these trustworthiness aspects from general AI to GNN systems,[1] and present an *open framework* (as shown in Fig. 1) on trustworthy GNNs. In practice, the characteristics of graph data differentiate the research with a specific focus on trustworthy GNNs from that exploring trustworthy AI. In Section I-C, we will introduce these core aspects in the context of GNNs, and present metrics and research differences related to each of them.

In this survey, we define **trustworthy GNNs** as *competent GNNs that incorporate core aspects of trustworthiness*, including **robustness, explainability, privacy, fairness, accountability, well-being, and other trust-oriented characteristics in the context of GNNs**. We believe that incorporating these aspects when designing GNNs will significantly improve their trustworthiness and promote their industrialisation and broad applications.

C. Aspects of Trustworthy GNNs.

From the view of deep learning, GNNs provide a generalised way to explore graph data. Unlike common image data, graph data is derived from non-Euclidean space [50] and has certain unique characteristics (e.g., discreteness, irregularity, non-IID nodes), that distinguish trustworthiness practices in GNNs from those in general AI. To better illustrate our trustworthy GNN framework, we here briefly explain each core aspect of trustworthiness, describe the metrics used for evaluation, and pinpoint the differences between research into trustworthy GNNs and general trustworthy AI (also summarised in Table II).

Robustness. Robustness refers to the ability of GNNs to remain stable under perturbations, especially those that are created by attackers. For example, a credit rating GNN in a financial system [51] should remain secure despite a perturbed

[1]The term "general AI" in this survey refers to artificial intelligence methods for exploring Euclidean space data [50] (e.g., images).

Fig. 1. The Open Framework of Trustworthy GNNs and Overview of Methodology Categorisation. Trustworthy GNNs are competent GNNs that incorporate aspects of trustworthiness, including robustness, explainability, privacy, fairness, accountability, well-being, and other trust-oriented aspects in the context of GNNs. For each aspect of trustworthy GNNs, we comprehensively summarise typical methods and provide methodology categorisation, as will be shown in the following sections.

transaction graph in which attackers attempt to forge connections between users with high credit scores.

Metrics. Generally, the robustness of GNNs can be measured with reference to the model accuracy [52], attack success rate of certain attack algorithms [28], mis-classification rate [53], structural similarity score [54], and attack budget [55].

Research Differences. Since graph structure is highly discrete, the notion of perturbations on graph samples (e.g., adding/deleting edges) is fundamentally different from those on ordinary samples in Euclidean space (e.g., images) [44].

Explainability.[2] Similar to general deep learning models, most GNNs are regarded as black-box models due to the absence of explainability. Explainability enables the predictions of GNNs to be traceable, which in turn enables humans to understand the behaviours of GNNs and discover knowledge. For instance, GNN explanations (e.g., subgraphs) can reveal which components in molecule graphs support the final biochemical functionality predictions of GNNs [56].

[2]We will expand on the difference between interpretability and explainability in Section IV-A.

TABLE II
TYPICAL METRICS FOR ASPECTS IN TRUSTWORTHY GNNS AND TYPICAL RESEARCH DIFFERENCES BETWEEN TRUSTWORTHY GNNS AND AI

Aspects		Robustness	Explainability	Privacy	Fairness	Accountability	Environmental Well-being
Typical Metrics		ASR*, Accuracy [28]	Accuracy [29], Faithfulness [30]	ASR*, Data Leakage [31]	Group/Individual Fairness [32],[33]	Standard Evaluations [34]	Inference Time [35], Nodes-Per-Joule [36]
Typical Research Differences	**Items**	**Perturbation**	**Explanation Form**	**Private Object**	**Data Distribution**	**Measurement**	**Efficiency Bottleneck**
	AI	Continuous-valued Perturbations [37]	Sub-images [38], Saliency Maps [39]	Individual Samples [40]	IID*[41]	Ordinary Metrics [42]	Model Scale [10], Energy-intensive Architectures [43]
	GNNs	Discrete-valued Perturbations [44]	Nodes, Edges, Subgraphs [29]	Edges [45], Nodes [46], Graphs [47]	Non-IID [33]	Graph-based Metrics [34]	Data Scale and Irregularity [48], [49]

* ASR indicates the attack success rate of adversarial/privacy attacks on robustness/privacy. IID stands for independent and identically distributed.

Metrics. Explanation accuracy [29], explanation faithfulness [30], explanation stability [57] and explanation sparsity [58] are commonly used metrics for evaluating GNN explanations. In specific applications, some metrics based on domain knowledge (e.g., correlated separability [59] in computational pathology) are also used to measure explanations.
Research Differences. Unlike explanations for general AI (e.g., saliency maps [39], sub-images [38]), GNN explanations (e.g., nodes, edges, subgraphs) [29] inherit the irregularity and discreteness of graph data, which go beyond the data characteristics in general AI (e.g., regularity of image data). Thus, the generation of GNN explanations requires specially designed methods [29].

Privacy. Privacy indicates how private data within GNNs can be protected to prevent it from being leaked. Specifically, the privacy of graph data and model parameters, which are regarded as confidential information belonging to their owners, should be guaranteed, and any exposure to unauthorised parties should be prevented [60], [45].
Metrics. Generally, a private GNN requires that no leakage of private data (e.g., nodes, edges, the graphs themselves, GNN model parameters, and hyper-parameters for GNN training) occurs in its systems [61]. Privacy can also be measured based on the ability of GNNs to defend against privacy attacks and reduce their attack success rates [62].
Research Differences. Graph data, as an important component of GNNs, imposes additional privacy requirements. In addition to protecting the sensitive information of the entities (i.e., nodes), the relationships between them (i.e., edges) also need to be protected [63].

Fairness. Discrepancies in predictions made with regard to marginalised groups indicate that the behaviour of some GNNs is unfair. Recent studies have shown that GNNs can discriminate against samples based on protected and sensitive attributes like gender and nationality (e.g., discrimination in credit risk prediction [64]). Fairness requires GNNs to accommodate differences between people and provide the same quality of service to users from diverse backgrounds.
Metrics. Currently, group fairness (e.g., dyadic fairness [32], demographic parity and equalized odds [65]), individual fairness (e.g., similarity-based fairness [33], ranking-based fairness [66]), and counterfactual fairness [64] are commonly used metrics for evaluating the fairness of GNNs.
Research Differences. Unlike the data distribution in general AI fairness, the existence of edges between nodes breaks the assumption that samples are independent and identically distributed. This implies that specific fairness definitions (e.g., involving a similarity function defined on edges) are necessary when studying fairness in GNNs for node-focused computational tasks (see Section II); examples include node classification [33] and link prediction [32].

Accountability. Accountability is another aspect of trustworthy GNNs that considers the ability to determine whether a system is performing in accordance with procedural and substantive standards [67]. Accountable GNNs contain a set of specific assessments and metrics to justify and identify the responsibilities associated with individual roles or development steps in GNNs.
Metrics. The accountability of GNNs can be measured by whether a standard evaluation process exists [34] and how well the specifications are established for each step or role of the GNN life cycle [42].
Research Differences. Due to the unique characteristics of graph data, the evaluation standards for GNNs differ from those of other deep learning methods. For example, graph datasets with different graph-based metrics (e.g., node degrees) are used to assess how GNN architectures perform differently on various datasets [34]. Therefore, researchers need to design new procedures and techniques specifically for building an accountable GNN.

Environmental Well-being. Well-being evaluates if GNNs are aligned to people's expectations regarding social good.[3] There is currently an urgent need to deploy GNNs on edge devices for real-time inference systems with limited processing

[3]Well-being is a generalised term, which encompasses both environmental well-being and various other forms of well-being (ranging from generating long-term value for shareholders to ending poverty [12]). In contrast to other forms of well-being, environmental well-being has mainly been explored from the technology perspective. Thus, in this survey, discussions of well-being will focus on introducing existing advancements in environmental well-being.

TABLE III
COMPREHENSIVE COMPARISON BETWEEN OUR SURVEY AND REPRESENTATIVE SURVEYS

Surveys	Scope			Perspective						
	Generic GNNs	Trustworthiness		Robustness	Explainability	Privacy	Fairness	Accountability	Environmental Well-being	Relations
		AI	GNNs							
Wu *et al.* [2]	●	○	○	○	○	○	○	○	○	○
Ruiz *et al.* [3]	●	○	○	○	○	○	○	○	○	○
Zhang *et al.* [68]	●	○	○	○	○	○	○	○	○	○
Liu *et al.* [10]	○	●	○	●	●	●	●	●	●	●
Li *et al.* [11]	○	●	○	●	●	●	●	●	○	●
Jin *et al.* [28]	○	◐	◐	●	○	○	○	○	○	○
Yuan *et al.* [29]	○	◐	◐	○	◐	○	○	○	○	○
Abadal *et al.* [69]	○	◐	◐	○	○	○	○	○	◐	○
Dai *et al.* [70]	○	◐	◐	●	●	●	●	○	○	○
Our Survey	○	◐	●	●	●	●	●	●	●	●

* In this table, ○ indicates "Not Covered", ◐ indicates "Partially Covered", ● indicates "Fully Covered".

power and memory resources (e.g., point cloud segmentation in autonomous vehicles [71]). As a representative sub-aspect of well-being, environmental well-being focuses on measuring the efficiency of GNNs.

Metrics. Generally, resource/efficiency-related metrics such as inference time [35], memory footprint [72], or nodes-per-joule [36] are employed to evaluate efforts on environmental well-being of GNNs.

Research Differences. The efficiency bottlenecks of general AI are caused by the scale [10] and energy-intensive architecture [43] of models. Due to the powerful presentation capacity and characteristics of graphs, the graph scale and data irregularity are the primary sources of the efficiency bottleneck in GNNs, whose operations rely on input graph structures [69].

Others. While we focus primarily on the above six aspects, which represent the mainstream studies, our proposed trustworthy GNN framework is an open framework that can readily incorporate other trust-oriented characteristics derived from principles outlined in trustworthiness guidelines [14], [15]. For example, trustworthy systems should be able to obtain knowledge from limited training data and achieve convincing competence on unseen data [11]. This requires GNNs to make compelling predictions on unseen real-world data, especially from domains or distributions that are distant from the training data [11], [73]. For these trust-oriented characteristics, definition-related metrics are employed to evaluate the extent to which GNNs conform to them. For example, one study on extrapolation [74] evaluates GNNs by calculating their prediction accuracy on test data, which lies outside the distribution of training data.

D. Related Surveys

Note that trustworthy GNN systems are also necessarily secure GNN systems. In computer systems, security refers to achieving confidentiality, integrity, and availability [75]. In the context of trustworthy GNNs, security is covered under the aspects of robustness, privacy and accountability (for example, defending against adversarial and privacy attacks, enabling accountability and verifiability in GNN systems, etc.).

Related surveys include reviews on GNNs and trustworthy AI, along with reviews on a single or multiple aspects of trustworthy GNNs (as shown in Table III). Compared with those surveys, our survey elaborates on existing advancements in the above six aspects of trustworthy GNNs and explores the complex relations between them.

Surveys on GNNs mostly focus on performance-oriented methods [2], [3], [68]. Wu *et al.* [2] review several different GNN architectures, including recurrent graph neural networks, convolutional graph neural networks, graph autoencoders, and spatial-temporal graph neural networks. Ruiz *et al.* [3] review GNNs from the perspectives of graph perceptrons, multiple-layer networks, and multiple-feature networks. Zhang *et al.* [68] categorise deep learning methods on graphs into graph recurrent neural networks, graph convolutional networks, graph autoencoders, graph reinforcement learning, and graph adversarial methods by distinguishing the basic assumptions/aims that underpin these methods.

Surveys on trustworthy AI review research into general AI systems [10]. Due to the unique characteristics of graph data and GNNs (e.g., the interdependence between nodes, intertwined computation between dense and sparse operations during GNN inferences [69]), these surveys do not appear to be able to thoroughly guide building trustworthy GNN systems. For example, since graph structure is highly discrete, the notion of perturbations on graph samples is intrinsically different from that of perturbations on ordinary samples in Euclidean space [44].

Surveys on a single aspect/multiple aspects of trustworthy GNNs have also emerged recently [28], [29], [69]. Jin *et al.* [28] review existing adversarial attack methods and corresponding defence strategies for GNNs. Yuan *et al.* [29] provide an overview of post-hoc explainers for GNNs by discussing the methodologies employed. Abadal *et al.* [69] summarise current software frameworks and hardware accelerators for GNNs. Note that these authors do not comprehensively consider how to build trustworthy GNNs. Some of them focus on subtopics of a particular aspect of trustworthy GNNs. For example, the survey [29] of explanation methods for GNNs mainly reviews post-hoc explainers.

Concurrent to our survey, Dai *et al.* [70] present an insightful review on trustworthy GNNs from four aspects, namely privacy, robustness, fairness, and explainability. Our survey differs from this work in three key ways. 1) *Consolidated and fine-grained taxonomy*. For each trustworthiness aspect, we provide a specific definition of the aspect, metrics for evaluation, and a fine-grained taxonomy. For instance, we categorise the defence methods for robustness into pre-training, during training, and post-training methods, enabling practitioners to build robust GNNs during different phases. 2) *Cross-aspect relations.* Our survey pinpoints the complex relations between aspects of trustworthy GNNs. In particular, our survey provides insights on a) *how the methods from one aspect of trustworthiness are adapted to address objectives in other aspects*, and b) *why advancing one aspect of trustworthy GNNs can promote or inhibit other aspects.* These relations are vital to a comprehensive understanding and construction of trustworthiness. 3) *Open framework*. Our survey provides an open framework (see Fig. 1) that includes six key aspects that have attracted increasing research attention in the field. This framework is extensible and thus capable of incorporating any aspect of trustworthiness that is yet to be considered but may be necessary for the future.

Table III presents a comparison between our survey and representative surveys on GNNs, trustworthy AI, and studies on one or multiple aspects of trustworthy GNNs. Taking a different approach from the above surveys, our survey thoroughly reviews current trust-oriented methods for constructing trustworthy GNN systems. We aim to provide a methodical roadmap for both researchers and industry practitioners working in the GNN community.

E. Contributions and Organisation

The contributions of our survey can be summarised as follows:

- **Open Framework for Trustworthy GNNs.** We propose an open framework containing trust-oriented characteristics for use in accurately characterising trustworthy GNNs. To illustrate the core aspects of this framework, we present their evaluation metrics along with the differences between research into trustworthy GNNs and that focusing on general AI.
- **Comprehensive Methodology Categorisation.** For each aspect of trustworthy GNNs, we comprehensively summarise the basic concepts and typical methods, and discuss future research directions.
- **Insights for Cross-aspect Relations.** We elaborate on the complex relations between the aspects in our open framework in terms of methodologies and achievements, which are vital to consider if fully trustworthy GNNs are to be achieved.
- **Outlook on Trending Directions.** By considering trustworthy GNNs as a whole and summarising the common limitations of current advancements, we point out several promising avenues of exploration that will need to be fully investigated to promote the research and industrialisation of trustworthy GNNs.

The remainder of this survey is organised as follows. We first introduce some key concepts related to graph neural networks in Section II. Working from the definition of trustworthy GNNs, we summarise current methods for improving the trustworthiness of GNNs from different aspects in Sections III-VIII, then discuss the complex cross-aspect relations in Section IX. Finally, we conclude this survey in Section X and point out trending directions of research into trustworthy GNNs by considering the concept as a whole.

II. Concepts

Graphs. A graph can be denoted as $G = \{\mathcal{V}, \mathcal{E}\}$, where $\mathcal{V} = \{v_1, \ldots, v_{|\mathcal{V}|}\}$ is a set of nodes, $\mathcal{E}$ is a set of edges. $\mathcal{E}$ describes the structural information of G, which can also be expressed as the adjacency matrix $\mathbf{A} \in \{0,1\}^{|\mathcal{V}|\times|\mathcal{V}|}$. $\mathbf{A}_{i,j} = 1$ if $e_{ij} = (v_i, v_j) \in \mathcal{E}$, otherwise $\mathbf{A}_{i,j} = 0$. The edge feature associated with edge e_{uv} is $\mathbf{e}_{uv}$. The node features are expressed as a matrix $\mathbf{X} \in \mathbb{R}^{|\mathcal{V}|\times d}$, whose i-th row indicates the feature values of v_i and d presents the dimensionality of features. Thus, a graph can also be expressed as $G = \{\mathbf{A}, \mathbf{X}\}$.

A subgraph $G_s = \{\mathcal{V}_s, \mathcal{E}_s\}$ of G is composed of a node subset $\mathcal{V}_s \subseteq \mathcal{V}$ and an edge subset $\mathcal{E}_s \subseteq \mathcal{E}$, where $\mathcal{V}_s$ contains all nodes involved in $\mathcal{E}_s$. Correspondingly, when features are associated with nodes, a subgraph can also be expressed as $G_s = \{\mathbf{A}_s, \mathbf{X}_s\}$.

Graph Neural Networks. GNNs are a series of neural network architectures designed to explore graphs by learning graph embedding [76], [77], [78]. We here introduce typical message-passing-based GNNs, in which the node embedding is iteratively updated as follows [76]:

$$\begin{aligned} \mathbf{m}_v^{(t)} &= \sum_{u\in\mathcal{N}(v)} M_t(\mathbf{h}_u^{(t-1)}, \mathbf{h}_v^{(t-1)}, \mathbf{e}_{uv}), \\ \mathbf{h}_v^{(t)} &= U_t(\mathbf{h}_v^{(t-1)}, \mathbf{m}_v^{(t)}), \end{aligned} \tag{1}$$

where $\mathbf{h}_v^{(t)}$ indicates the embedding of node v at layer $t \in \{1, \ldots, T\}$, and $\mathcal{N}(v)$ denotes the set of neighbours of v in graph G. $M_t(\cdot,\cdot,\cdot)$ and $U_t(\cdot,\cdot)$ are the message function and the embedding updating function at layer t, respectively. By means of the above operations, message-passing-based GNNs can provide graph embedding for various tasks.

Computational Tasks. Tasks on graphs can be categorised into node-focused tasks and graph-focused tasks [51]. *Node-focused tasks* mainly include *node classification, node ranking, link prediction, and community detection* [79]. *Graph-focused tasks* mainly include *graph classification, graph matching, and graph generation* [51]. Two representative tasks are discussed in more detail below.

Node classification. Given a graph $G = \{\mathcal{V}, \mathcal{E}\}$, $\mathcal{V}_l \subset \mathcal{V}$ denotes the set of labelled nodes, and the label associated with $v_i \in \mathcal{V}_l$ is y_i. $\mathcal{V}_u = \mathcal{V} \setminus \mathcal{V}_l$ is the set of other unlabelled nodes. The goal of node classification is to learn a GNN model f_θ from G and node labels, then utilise f_θ to predict labels for the nodes in $\mathcal{V}_u$.

Graph classification. Given a graph dataset $\mathcal{D} = \{(G_i, y_i)\}$, in which y_i is the label of graph G_i, the goal of graph classification is to learn a GNN model f_θ from $\mathcal{D}$, then use f_θ to predict labels for unseen graphs.

More details about other GNN architectures (i.e., architectures beyond the message-passing mechanism) and tasks (e.g., link prediction, graph generation, etc.) can be found in several books [51], [80] that provide an introduction to GNNs.

III. Robustness of GNNs

Robustness is one of the most critical aspects of trustworthy GNN development. In general terms, robustness refers to the ability of systems to perform persistently under a variety of conditions. In the context of GNNs, a robust GNN can sustain model accuracy under perturbations such as malicious graph structure modifications made by adding or deleting edges. Recent studies [81], [82], [83] have demonstrated that GNNs may output inferior results when graph structure is perturbed. For example, when targeting a GNN used for malware detection, attackers can insert/delete methods or add call relations into their original malware (i.e., inserting/deleting nodes and adding edges), so as to change the function call graph of the malware and bypass detection [84]. These security risks are becoming increasingly prominent when GNNs are used for such critical applications. Thus, it is imperative to study adversarial attacks in order to fully understand the risks of existing GNN systems and develop proper defence strategies to improve their robustness.

In this section, we summarise recent studies on GNN robustness. Ideally, a robust GNN should be capable of remaining stable under circumstances of both adversarial attacks and random errors (e.g., unexpected omissions in the graph data [85]). While research has shown that random failures are often less severe [86], most current studies focus on robustness when dealing with adversarial attacks [81], [82], which is critical due to its detrimental impacts on GNN-based applications. Therefore, we will focus more on robustness against adversarial attacks. We will first introduce some basic concepts regarding attacks and defences. Subsequently, we will introduce representative attack methods along with common defence mechanisms. We will also discuss future directions at the end of this section.

A. Adversarial Attacks

1) Threat Models and Attack Categories: Threat modeling represents how an attacker might use adversarial perturbations to disrupt the performance of a GNN model. There are several aspects in the threat model of GNN adversarial attacks, including when an attack takes place, what information an attacker can obtain, what adversarial actions can be taken, and how the attacker might attempts to harm the performance of the targeted GNN model. In this section, we will introduce different types of attacks based on their threat models. Fig. 2 illustrates the overall processes of several common attacks. The dashed-line boxes in the lower left show how GNNs are trained and used in a clean graph without adversarial attacks. The three dashed-line boxes in the upper left corner (from top to bottom) illustrate how evasion attacks, poisoning attacks, and backdoor attacks are applied to GNN training and/or inference phases. Within these dashed-line boxes, the components affected by adversarial actions are highlighted in red, while those not impacted by such actions appear in blue. The upper right corner contains a dashed-line box that presents the attack goals. Next, we will introduce the attacks in GNNs by presenting their attack stages, attacker knowledge, perturbation objects, and attack goals.

Attack Stages. Attacks can occur either during the development phase (aka training) or the deployment phase (aka inference). Generally, these attacks can be divided into two categories: *poisoning attacks* and *evasion attacks*.

- *Poisoning attacks.* Poisoning attacks target the training phase of GNN model development. Attackers attempt to alter the training graphs of a target GNN, leading to the production of a poisoned model with impaired performance [87], [88], [89].
- *Evasion attacks.* Attacks that target the inference phase are known as evasion attacks. Attackers aim to perturb the graphs during the inference period so that the perturbed graph can cause the clean GNN models to engage in incorrect behaviours (e.g., misclassification) [52], [90], [91]. Different from poisoning attacks, attackers in evasion attacks cannot affect the parameters of the target GNN models. Therefore, it is assumed that the target GNN models have been well-trained on the clean graph and are considered to be fixed during this type of attacks.

There is another type of attack, namely a backdoor attack, which perturbs both the training and inference graphs [92]. Specifically, attackers inject backdoors as pre-defined conditions (e.g., an input graph consisting of a specific subgraph) into GNN models by poisoning the training graph. Subsequently, during inference, attackers can perturb the inference graph to trigger the backdoor and force the target GNNs to misbehave.

Attacker's Background Knowledge. In practical scenarios, an attacker may be able to gather different types of knowledge. This knowledge typically consists of the target GNN model (e.g., model parameters, architectures), and the training or testing data. Based on the type and degree of knowledge obtained, attacks can be classified into three categories: *white-box attacks*, *black-box attacks* and *grey-box attacks*.

- *White-box attacks.* In a white-box attack, attackers have access to the entire target GNN, including its architecture, parameters, and gradient [82], [93]. This information can be used to construct adversarial graphs; for example, attackers may seek the desired perturbations by calculating the gradient corresponding to prediction errors (see details in Section III-A2). Note that white-box attacks require the strongest assumption with regard to knowledge of the target system, which may be not available in real-world applications.
- *Black-box attacks.* By contrast, a black-box attack means that attackers do not know anything about the target GNN model, and can only send queries to GNNs [89], [52]. A common approach [44], [52] to conducting a black-box attack involves building a surrogate model. Specifically, attackers manage to gather information from the target GNN model (e.g., exploring the mapping between designed input queries and their responses) to construct a surrogate model.

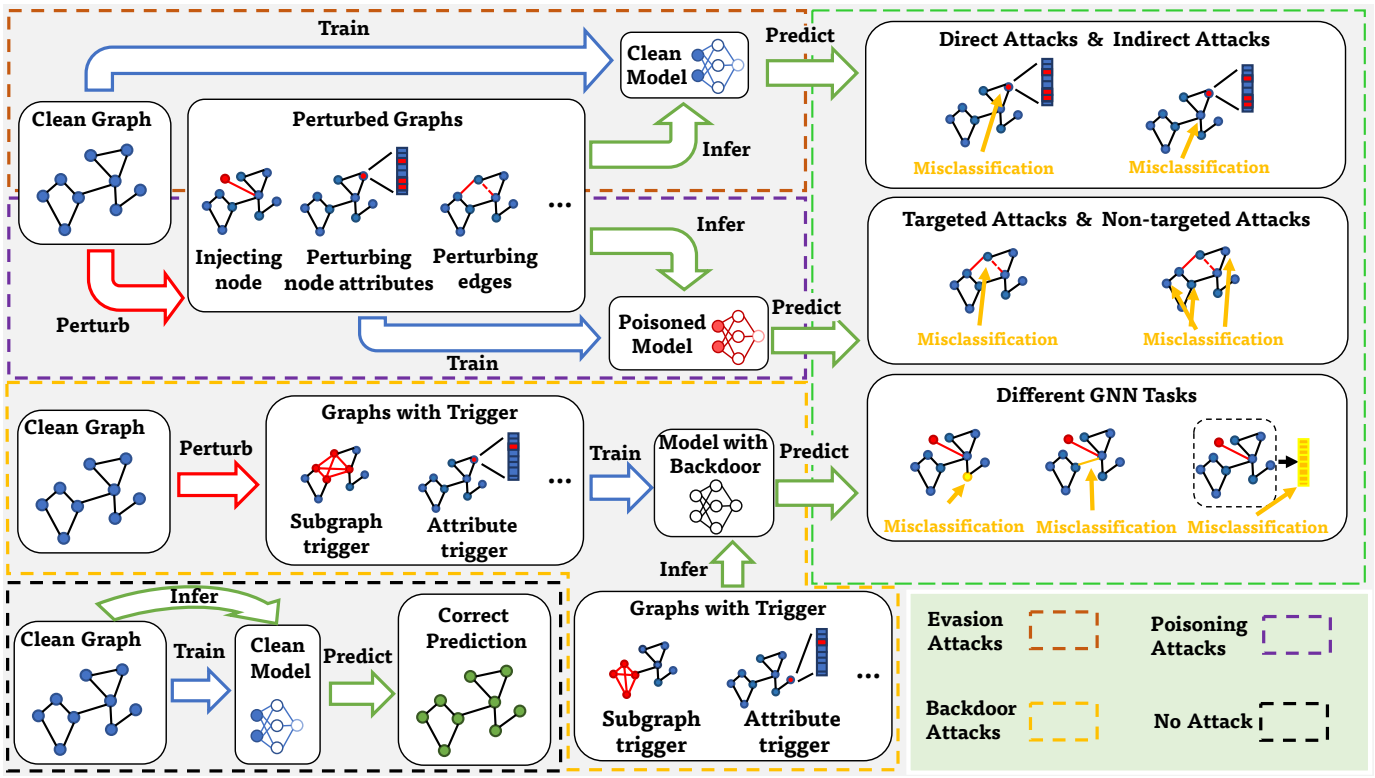

Fig. 2. Process of different adversarial attacks against GNNs. Training and inference without adversarial perturbations can produce a clean GNN model and correct prediction results (as shown in the black dotted box). Meanwhile, different perturbations to the clean graph are added during either a training or inference process to produce evasion attacks (brown box), poisoning attacks (purple box), and backdoor attacks (yellow box), which lead to different kinds of misclassifications (green box).

Then, with full access to the surrogate model, attackers can utilise white-box attack strategies to generate adversarial graph samples. It is worth noting that compared with white-box attacks, black-box attacks are more dangerous, since they are more applicable to real-world scenarios.

Perturbation Operations. An attacker's capability is also defined in the threat model. Most existing attacks propose to inject perturbations into graph data. Specifically, there exist multiple types of perturbations, including the following:

- *Perturbing graph structure.* Most existing studies propose to implement their attacks by modifying edges (i.e., adding/deleting/rewiring edges) [52], [82], [94]. In practice, attackers often constrain their perturbations within a given budget (e.g., making modifications to only a limited number of edges), which makes them more difficult to detect.
- *Perturbing node attributes.* In addition, perturbations can also be added to node attributes [95]. Notably, empirical studies have shown this approach to be less effective than perturbations on edges [44], [96]; consequently, it is often used in combination with other types of perturbations [44], [87].
- *Injecting nodes.* Another practical perturbation is to inject malicious nodes into graph data [89], [81], [93]. These malicious nodes can be linked to benign nodes, leading to misclassifications. This type of perturbations is attractive when attackers cannot modify existing edges. For example, it is difficult for attackers to break a credit ranking system and tamper with existing connections, but easier for them to register a new account and perform malicious actions like adding new connections with other accounts.

Attack Goals. Attack goals describe how attackers expect the performance of GNN models to be reduced. In general, they can be described from two perspectives.

- *Error specificity.* Attackers may expect different types of errors to result from their adversarial attacks. For example, in a node/graph classification task, an error-specific attacker expects the target GNN to misclassify nodes/graphs under a specific label, while error-nonspecific attackers only expect misclassifications more generally without specific target labels.
- *Attack specificity.* Attackers may have different attack targets. Specifically, in node-level GNNs, some attackers [44] expect only a small set of nodes to be misclassified, while others aim to degrade the overall performance of the target GNNs [87]. It should be noted that existing studies [44], [87] of GNNs also refer to the former as targeted attacks and the latter as non-targeted attacks. This convention differs from that used for attacks in non-graph fields (e.g., images), where misclassifying inputs under a specific label is known as a targeted attack, while misclassifying inputs to any incorrect label is known as a non-targeted attack.

2) Attack Methods: In this section, we will describe several representative attack methods.

Attack Formulation. To perform the above attacks, attackers

formalise the construction of perturbations as one of several optimisation problems. Specifically, attackers aim to find a perturbed graph, that can reduce the performance (e.g., overall accuracy on the testing set) of GNN models. The objective functions and constraints of the formalised optimisation problem can be determined with reference to the aspects introduced in Section III-A1. Here, we provide an example of the problem formalisation from Nettack [44]: given a target graph $G = \{\mathbf{A}, \mathbf{X}\}$, and a target node v_i with ground truth label y_i, an attacker attempts to construct a perturbed graph $\hat{G}$ with limited perturbations, such that the GNN model f_{θ^*} trained on $\hat{G}$ undergoes the maximum performance drop for the node classification task.

$$\begin{aligned} \max_{\widehat{G}=\{\widehat{\mathbf{A}},\widehat{\mathbf{X}}\}} \quad & \mathbb{I}(f_{\theta^*}(\widehat{G}, v_i) \neq y_i) \\ s.t. \quad & D(\widehat{G}, G) < \epsilon, \\ & \theta^* = \arg\min_{\theta} L(\theta; \widehat{G}), \end{aligned}$$

where $\mathbb{I}(\cdot)$ is a binary indicator function, $D(\cdot, \cdot)$ is a perturbation measurement function to assess the distance between the clean graph G and the perturbed graph $\widehat{G}$, and ϵ is a perturbation budget.

Considering that Nettack is a poisoning attack (it can also be adapted as an evasion attack), it directly perturbs the training graph from G to $\widehat{G}$, so as to generate the target GNN $f_{\theta^*}(\cdot, \cdot)$. In addition, the attacker simultaneously perturbs the graph structure and node attribute values, so that the generated perturbed graph $\widehat{G}$ includes both of these perturbations $\{\widehat{\mathbf{A}}, \widehat{\mathbf{X}}\}$. Finally, it aims to cause misclassification of a specific node; thus, the objective function of the attack is designed to measure the error between the prediction of $f_\theta(\widehat{G}, v_i)$ and its ground truth label y_i.

Optimisation Solving. As a next step, attack methods are proposed via solving optimisation functions. Two main strategies are employed to achieve this: namely, *gradient-based* and *non-gradient-based methods*.

- *Gradient-based methods.* Since white-box attacks are able to utilise information about gradients, attackers can employ gradient-based methods [82], [93]. Specifically, by regarding the desired perturbed graph as a hyperparameter, attackers can calculate the partial derivative of the loss (e.g., the loss between the predictions and ground truth labels) with respect to input graphs. These optimisations can be performed using gradients that are very similar to the training gradients [97]. Nevertheless, the difficulty lies in the discreteness of graph data. There are several approaches to solving such a problem [52], [98], [99]. One common method is to greedily and iteratively perturb the graph based on the gradient [100], [52]. Alternatively, the attacker can use techniques such as integrating gradients, among others, to convert the discrete data to continuous data.
  For example, the integrating gradients of a loss function with respect to an edge perturbation on the input graph are represented as follows [99]:

$$\begin{cases} \frac{\mathbf{A}_{ij}}{m} \times \sum_{k=1}^{m} \frac{\partial f_\theta(\frac{k}{m} \times (\mathbf{A}_{ij} - 0))}{\partial \mathbf{A}_{ij}}, \text{removing edges} \\ \frac{1-\mathbf{A}_{ij}}{m} \times \sum_{k=1}^{m} \frac{\partial f_\theta(\mathbf{A}_{ij} + \frac{k}{m} \times (1-\mathbf{A}_{ij}))}{\partial \mathbf{A}_{ij}}, \text{adding edges}, \end{cases}$$

  where $\mathbf{A}$ is the adjacency matrix, $\mathbf{A}_{ij}$ specifies the perturbed edges, and m is the step number for computing integrating gradients. In this case, integrating gradients can be regarded as the importance scores for edge perturbations, where a higher value indicates that the perturbation will result in better attack effectiveness. After calculating all the gradients, the attacker can greedily find the perturbation that produces the final adversarial graph.
- *Non-gradient-based methods.* In black-box attack scenarios, however, the gradients are unknown to attackers. There are several approaches [52], [101], [89] designed for solving the optimisation problem without using gradients, such as reinforcement learning [89] and genetic algorithms [52].
  Specifically, attackers utilising reinforcement learning algorithms can define graph perturbations as executing actions, then design rewards based on their attack goals [52]. The attack procedure can be modelled as a Finite Horizon Markov Decision Process (FHMDP): 1) *Action (a).* A single action at time step t is denoted as a_t, which represents adding or deleting an edge; 2) *State (s).* The state is represented as a tuple $(\widehat{G}^t, u_i)$, where $\widehat{G}^t$ is the perturbed graph at time step t and u_i is the attacker's target node; 3) *Reward (r).* The attacker receives no reward in the intermediate steps, i.e., $\forall t = 1, 2, ..., m-1, r(s_t, a_t) = 0$. Meanwhile, the attacker receives a non-zero reward at the end of FHMDP:

$$r(s_m, a_m) = \begin{cases} 1 & \text{if } f_\theta(\widehat{G}^m, u_i) \neq y_i \\ -1 & \text{if } f_\theta(\widehat{G}^m, u_i) = y_i. \end{cases}$$

  4) *Terminal.* The process terminates after modifying m edges.
  Furthermore, genetic algorithms can also be used to select the population (i.e., a perturbed graph) based on a fitness function that iteratively evaluates attack effectiveness [52].

Comparison. In addition to attack stages, perturbation objects, and attack goals that affect the formulation of the attack, optimisation solving methods are determined by the degree of attacker knowledge. White-box attacks, for example, use gradient information to construct their perturbations, while black-box attacks make use of techniques other than gradients, such as greedy search or reinforcement learning. Gradients offer the advantage of being straightforward and easy to implement. A number of works use this approach as a baseline, since it illustrates the vulnerability of victim models in the worst-case scenario [102].

In practice, however, an attacker may not have access to the gradients of the victim models. Thus, rather than utilising the gradients of the victim models directly, some black-box attacks [81] construct surrogate models and use the gradients obtained therefrom. Another study in black-box settings [103] uses adversarial graph search and query responses of the victim models to calculate the sign gradients. These approaches may require additional effort to determine the gradients (e.g., the time cost for the Topology attack using a surrogate model can be six to ten times that of a simple gradient-based approach [104]). Meanwhile, non-gradient methods are also widely employed in many studies [52], [101]. Most of these methods do not require knowledge of gradients, which makes

them more practical when dealing with situations in which an attacker has limited knowledge.

B. Defences

The methods employed to defend against the above attacks can be classified based on the targeted phases of GNN systems, i.e., defences occurring before, during, and after training.

1) Defences before GNN Training: Since most attacks tend to perturb the graph data, it is natural to consider preprocessing the graph data before training.

Training Graph Preprocessing. Graph preprocessing aims to distil current graph data by removing potential adversarial perturbations. As studies [105], [99] have shown, many perturbations applied by attackers have specific characteristics. By referring to these characteristics, defenders can identify suspicious components and remove them before GNN training begins. Defenders can also compute the low-rank approximation of the adjacency and feature matrices then discard any high-rank perturbations [105].

Clean Graph Learning. These types of defences propose to directly regenerate a clean graph rather than deleting a small number of suspicious edges. The intuition supporting the use of graph learning techniques in these scenarios is that, adversarial perturbations normally conflict with the basic graph property [106]. Thus, graphs reconstructed using these graph structure learning methods can erode the negative effects of the adversarial perturbations [99]. In addition, graph sanitation has also been employed by learning a "better" training graph from a perturbed graph [107].

2) Defences during GNN Training: Alternatively, some defence methods aim to improve the GNN robustness during the training process.

GNN Architecture Optimisation. A large number of studies have proposed to improve GNN robustness by adjusting GNN architectures. For example, a method called Gaussian-based GCN [108] introduces Gaussian distributions into graph embedding rather than using them directly. Moreover, defenders can use a graph encoder to learn the information, followed by a GAN model to conduct contrastive learning [109]. Alternatively, they can introduce meta-optimisation and adapt the aggregation process or completely redesign the GNN architecture [110].

Attention Mechanism. Another effective approach involves introducing attention mechanisms into GNNs [111] to reduce the impacts of adversarial edges or nodes while maintaining the performance on the clean graph. Specifically, these methods utilise the attention scores to identify adversarial edges and nodes, then decrease their weights during the aggregation process of the GNN training. For example, a defender could assign higher scores to edges connecting more similar nodes [111].

Adversarial Training. Adversarial training enhances the robustness of a model by introducing noise during GNN training [112], [113]. Specifically, defenders first use existing attack methods to construct perturbed graphs (noise) then train GNNs on these perturbed graphs to counteract the negative impacts of potential adversarial attacks [98]. Consequently, this method is more effective in dealing with specific attacks, whose adversarial samples can be used in training, when compared to other defensive methods. Furthermore, since only training graphs are modified under this approach, GNN developers do not have to modify the GNN architecture, meaning that the implementation effort required is relatively small. Nevertheless, it may be less generalisable to other types of attacks, especially those that are unknown to the developer, since adversarial samples from these attacks have not been used during GNN training.

Robustness Certification. Robustness certification can provide certification of a GNN's robustness under certain perturbations [114]. Specifically, it is used to quantify how node predictions from GNNs can be robust to arbitrary forms of perturbations that are bounded in a considered space. Once a node is certified, robustness certification guarantees that perturbations within the bounded space cannot change the GNN's prediction. Recently, certification methods have been used to perform robust training of GNNs [114], [115], [116]. Defenders add an extra term to the objective function of the target GNN model, which aims at driving every node to be certified. As a result, the trained GNNs can be more robust to perturbations.

3) Defence after GNN Training: Defence mechanisms can also be implemented after the GNN training. Consequently, such defences can be used to defend against evasion attacks that occur during the GNN inference phase.

Detection of Malicious Perturbations during Inference. Another common defence approach is to detect the malicious perturbations. Approaches of this kind explore the difference in inference performance between the attacked GNNs and the normal GNNs. Specifically, a recent study [117] has shown that, the output scores of the malicious node predictions are significantly lower than those of clean nodes. Therefore, defenders can utilise this evidence to identify adversarial attacks. In addition, robustness certification can also be used to identify the potential target nodes by analysing their robustness levels [114], which can help defenders to detect perturbations.

4) Summary: While a growing number of methods have been designed to enhance the robustness of GNNs, they are applied in different scenarios. Below, we compare the methods in terms of several aspects and discuss their application scenarios.

- *Stage of application.* These defence methods are designed to counter attacks that target different stages of GNNs. As an example, pre-training defence methods should be implemented prior to training. As a result, they will be better able to defend against poisoning attacks that perturb the training graph. However, if the perturbation occurs during the inference period (i.e., evasion attacks), these approaches cannot counteract it. By contrast, post-training defences can detect perturbations during the inference process; however, they are unable to deal with poisoning attacks in which an inaccurate GNN model has been built and deployed for inference. Meanwhile, the during-training defence strategies [108], [111] may be able to counter both types of attacks (poisoning and evasion).

- *Modularity.* Each of these defence methods has a distinct modularity. Consider an existing GNN development pipeline that needs to be updated to a robust one. When using a pre-training or post-training defence, there is no need to alter any existing process; these methods can be implemented in a plug-and-play fashion, which can easily be incorporated into the current workflow. For during-training defences, however, it may be necessary to make changes to the training programs.
- *Deployment compatibility.* Defence methods also suffer due to varying levels of compatibility during deployment. Specifically, pre-training defences can generate robust GNNs during their development; thus, the deployment of these GNNs does not need to be modified in comparison to the normal GNN deployment. In the meantime, during-training defence methods may require specific GNN architectures. For instance, the RGCN [108] should model the hidden layer representation of nodes as Gaussian distributions and exploit them during information propagation to reduce the impact of adversarial perturbation. Furthermore, a post-training defence method may require the GNN systems to provide the function block for detection (e.g., comparing the output scores to a threshold [117]) in GNN systems. Because these methods may require specialised GNN architecture or additional function blocks that may not be supported by general frameworks, they are less generalisable.

C. Future Directions of Robust GNNs

Robustness Evaluations. Despite the increasing number of studies that have proposed to develop robust GNNs, mathematically demonstrating the robustness of GNNs is a challenging proposition. In general, most robustness evaluations of GNNs are based on their defensive ability, as GNN algorithms are difficult to interpret. For example, current robust GNN techniques assess their methods by using *attack deterioration*, which is the decreased accuracy after an attack compared to the accuracy without attack [112], [106], [105], and *defence rate*, which compares the attack success rate with or without defence strategies [118]. However, the robustness conclusions obtained by these metrics are based on the specific perturbations (e.g., perturbations generated by specific adversarial attacks), meaning that general robustness against other attack algorithms cannot be guaranteed. A robustness certification attempts to offer a formal certification of robustness, but it is affected by the architecture of GNNs and the type of graph data [114], [115]. This introduces barriers when evaluating newly developed robust GNN techniques and comparing them to existing studies. It is therefore essential to develop a quantitative metric for evaluating the robustness of GNNs (e.g., one that identifies the upper bound of the model's prediction divergence in a given domain [119], [120]), which can be applied uniformly across various GNN architectures, graph data, and different attack and defence methods.

Defence Scalability. Although several techniques for building robust GNNs have been developed, most of them have only been evaluated on medium-sized graphs and have not been applied to large-scale graphs in practice. This highlights the necessity of studying GNN robustness with high scalability. Researchers have recently demonstrated that GNNs in large-scale graph data can also be vulnerable to adversarial attacks [121]; furthermore, they have also demonstrated that previous attacks and robust GNN schemes are typically not scalable due to the large overhead incurred during training and inference [122]. One example concerns a revised aggregation technique [123] for robust GNN training, which involves summing up the distance matrix of neighbour node embeddings. Such an approach would increase the training effort required for large graph training [121]. In view of the wide variety of attack setups and GNN applications in existence, it is clear that the robustness analysis of GNNs applied to large-scale graphs is yet to be fully explored. It is accordingly crucial to develop more efficient and scalable defence techniques to support the building of robust GNN systems for real-world applications.

IV. Explainability of GNNs

The explainability of GNNs refers to the ability to make the predictions of GNNs transparent and understandable. People will not fully trust GNNs if the predictions they make cannot be explained; this lack of trust will in turn limit their usage in crucial applications associated with fairness (e.g., credit risk prediction [64]), information security (e.g., chip design [124]) and life security (e.g., autonomous vehicles [71], protein structure prediction [125]). Consequently, building trustworthy GNNs requires insights into why GNNs make particular predictions, which has driven an increase in research into the interpretability and explainability of GNNs. These abilities enable researchers to capture causality in GNNs [126] or insights for further investigation in applications [127], foster the implementation of robust GNN systems by developers [128], and guide regulators to ensure the fairness of GNNs [129].

In this section, we summarise current advancements in providing explanations for GNN predictions. We first introduce the basic concepts and categories with respect to the interpretability and explainability of GNNs. We then present some typical methodologies and conduct comparisons between them. Finally, we highlight some potential directions for future research.

A. Concepts and Categories

Interpretability and Explainability. Methods designed to promote machine learning interpretability can be categorised into intrinsically interpretable models and post-hoc explanation methods [130]. In the context of GNNs, the *interpretability* of GNNs utilises intrinsically interpretable designs in GNN architectures to explain predictions of GNNs. In contrast, the *explainability* of GNNs aims to provide post-hoc explanations after the training of GNNs. In this survey, the former kind of GNNs are called *interpretable graph neural networks*, and the latter kind of methods for explainability of GNNs are called *explainers for graph neural networks*.

Forms of Explanations. The types of explanation results for general machine learning models include feature summary statistics, feature summary visualisation, internal parameters,

data points (e.g., the identification of prototypes of predicted classes), and intrinsically interpretable (surrogate) models [130]. Although there are various differences between current methods designed for GNNs, the shared concern is which edges, nodes, or features are more important to the final outputs. Thus, the *explanation results* of GNNs are generally expressed as graphs with certain components (i.e., edges, nodes or features) highlighted as explanations for the current sample. Sometimes explanations of GNNs are presented in the form of subgraphs, since subgraphs with specified patterns (i.e., motifs) are the building blocks of some more complex networks [131].

Instance-level Explanations, Group-level Explanations, Class-level Explanations. Existing methods are categorised based on whether they provide instance-level, group-level, or class-level explanations for GNNs. The *instance-level* methods (e.g., PGExplainer [56]) provide a sample-dependant explanation for each graph sample. The *group-level* methods (e.g., GNNExplainer [22]) generate or choose a prototype sample as the explanation for a group of graph samples. The *class-level* methods use graph patterns (e.g., XGNN [132]) or statistics (e.g., OFS+OBS [127]) to explain the outputs of GNNs for specified classes. Compared with group-level and class-level methods, the explanations provided by instance-level methods are both more fine-grained and more precise since explanations for a single sample are tailored to explain the sample in question. In contrast, the class-level explanations provide a high-level insight into the behaviours of GNNs.

Model-specific versus Model-agnostic Methods. Explanation methods for GNNs can be further categorised as either *model-specific* or *model-agnostic* based on their design. Generally, interpretable GNNs (e.g., SAN [133]) tend to be model-specific; this is because explanations are derived from specifically designed operations in GNNs, which are used to reflect the importance of certain graph components (e.g., edges, nodes or features). Accordingly, these interpretable GNNs can usually only provide explanations for the proposed graph neural architectures. For their part, model-agnostic methods (e.g., XGNN [132]) can provide post-hoc explanations for any GNN. However, not all post-hoc explainers for GNNs are model-agnostic. The reason is that, like interpretations [134] for piecewise linear neural networks, some explainers (e.g., CAM [58]) are not rigorously model-agnostic, since they require GNNs to employ some specified operations (e.g., global average pooling [58]).

Others. Methods for explaining GNNs also differ depending on the *target task* of the GNNs in question. For interpretable GNNs (e.g., SAN [133]), they can only provide explanations for current architectures designed for specified tasks. For post-hoc explainers of GNNs, the difference on target tasks means that some explainers (e.g., GraphLime [135]) can only provide explanations for GNNs in one specified task (e.g., node classification), while others (e.g., RCExplainer [136]) are available for multiple GNN tasks. Moreover, existing methods have different needs in terms of the *prerequisite knowledge* of GNNs. In this survey, the terms white-box, grey-box, and black-box are used to represent the degree of knowledge required by different methods about the target GNNs when providing explanations for them. Some explainers [132] (i.e., black-box methods) treat the GNNs as black boxes, while others (e.g., white/grey-box methods) may need to access some internal information (e.g., the backward gradients [22], weight parameters of GNNs [58]) from GNNs to train explainers.

B. Methods

When exploring methods for explaining GNNs, two essential steps are designing the method for providing explanations and evaluating explanations. Both of these steps can be challenging for researchers. For example, due to the characteristics of graph data, traditional explanation methods for deep learning (e.g., input optimisation methods [137] and soft mask learning methods [138]) cannot be directly applied to GNNs because of the irregularity and discrete topology of graphs. When conducting evaluations, domain knowledge (e.g., brain connectomics [127]) is sometimes necessary to validate GNN explanations.

To this end, various intrinsically interpretable GNNs and post-hoc explainers for GNNs have been proposed. Here, we briefly introduce a typical post-hoc explainer for GNNs, i.e., GNNExplainer [22]. Taking the graph classification task as an example, we here assume that a well-trained GNN model f_θ is employed as a label function to produce the predicted label $\hat{y}$ of the input graph $G = \{\mathbf{A}, \mathbf{X}\}$, whose ground truth label is y. GNNExplainer aims to find a subgraph $G_s = \{\mathbf{A}_s, \mathbf{X}_s\}$ of G that can act as the explanation of prediction $\hat{y} = f_\theta(G)$ by maximising the mutual information ($\mathbf{MI}$) between G_s and y, i.e.,

$$\max_{G_s} \mathbf{MI}(y, G_s).$$

To solve the optimisation problem on the discrete graph structure, GNNExplainer proposes to learn an edge mask $\mathbf{M} \in \mathbb{R}^{|\mathcal{V}|\times|\mathcal{V}|}$ ($|\mathcal{V}|$ is the number of nodes in G), and the explanation G_s is calculated as follows:

$$G_s = \{\mathbf{A}_s, \mathbf{X}_s\} = \{\mathbf{A} \odot \sigma(\mathbf{M}), \mathbf{X}\},$$

where σ denotes the sigmoid function, and $\odot$ represents element-wise multiplication. After the training, GNNExplainer can generate an explanation for the prediction of GNN model f_θ.

In this section, we present common ideas behind different methods for GNN explanations. Fig. 3 illustrates an overview of the current methodologies. The bottom (i.e., green zone) of Fig. 3 shows four types of intuitions used in constructing self-interpretable GNNs. The other part (i.e., grey zone) of Fig. 3 illustrates the core modules used in five different kinds of post-hoc explainers and how they interact with target GNNs (green dashed-line box) to generate explanations. Moreover, it also illustrates other typical method instances that can provide post-hoc explanations for GNNs. In the below, we will introduce these methodologies and some typical instances of them.

1) Intrinsically Interpretable GNNs: The main types of self-interpretable GNNs mainly include contribution estimation, the introduction of interpretable modules, embedding prototype learning, and rationale generation.

Contribution Estimation. Contribution estimation methods

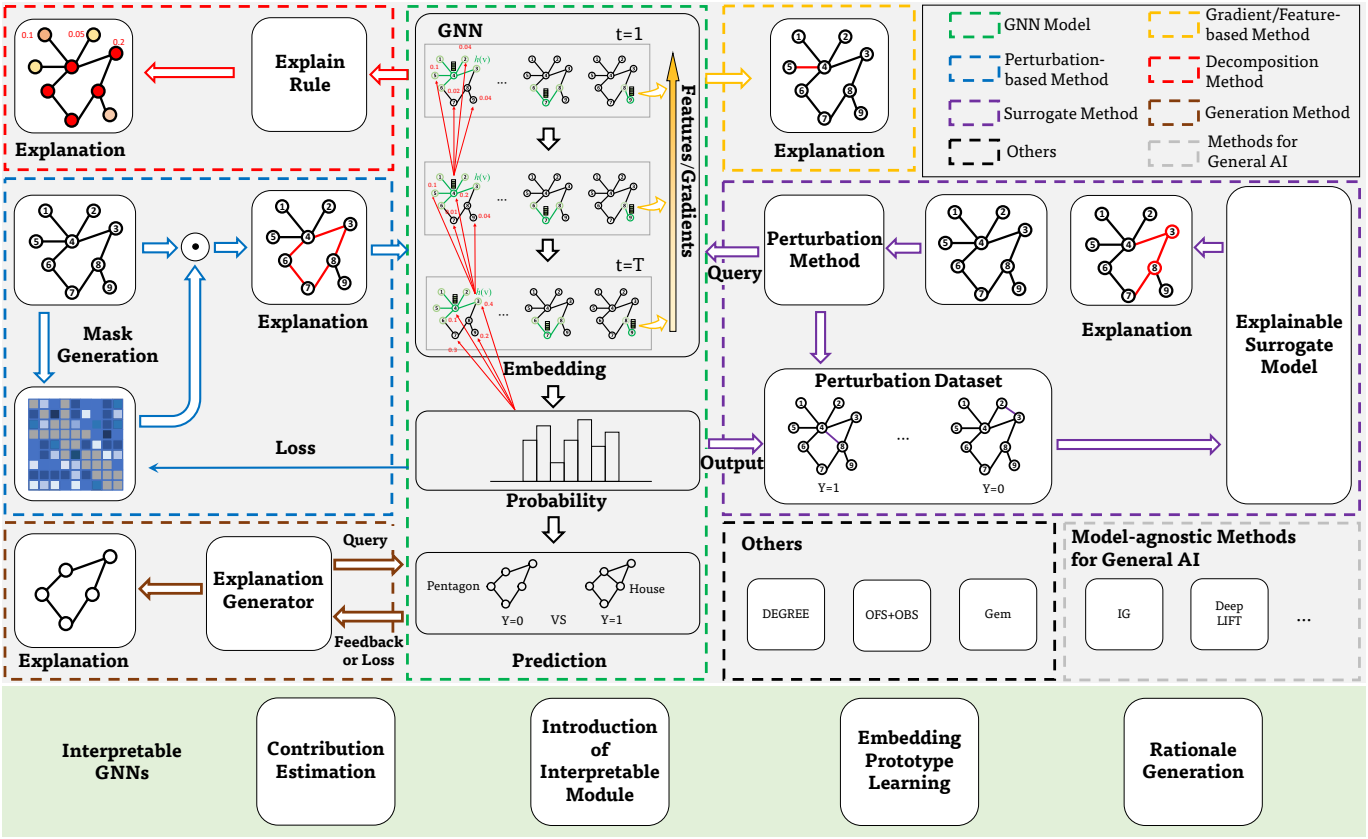

Fig. 3. An overview of methodologies for improving the interpretability and explainability of GNNs. Self-interpretable GNNs (green background) mainly include contribution estimation, introducing interpretable module, embedding prototype learning, and rationale generation methods. Explainers of GNNs (grey background) include gradient/feature-based methods, perturbation-based methods, surrogate methods, decomposition methods, generation methods, and other methods.

employ values from modules or operations in GNN architectures to evaluate the contribution made by particular components in input samples. These methods then utilise the estimated contributions to obtain explanations for predictions made by GNNs. In contribution-based methods, the values of contribution estimations can be derived from different modules or operations. For example, these contribution values can be provided by activation values (e.g., NGF [139]), attention weights (e.g., STANE [140]) or specially designed modules (e.g., SAN [133]).

Introduction of Interpretable Module. Interpretable models like K-Nearest Neighbours (KNN) can be introduced as a module in GNNs. The subsequent improvement to GNN interpretability mainly comes from changing the classification module (e.g., MLP) to an interpretable module like KNN, given that GNNs serve as encoders of graph data. This kind of architecture combines the powerful expressive ability of GNNs and the interpretability of existing interpretable methods. For example, a GNN architecture for node classification called SE-GNN [141] proposes identifying the interpretable k-nearest labelled nodes for each node to simultaneously make a label prediction and explain why this prediction has been made.

Embedding Prototype Learning. Rather than employing interpretable entities (e.g., labelled nodes in SE-GNN [141]) as in the introducing interpretable module methods, the interpretability of prototype learning methods comes from the learned embedding prototypes or embedding motifs in GNNs. For example, Zhang *et al.* [142] propose a framework called PortGNN, which incorporates prototype projection and similarity calculation operations. The interpretability of PortGNN on a specific input graph comes from displaying the learned prototypes and similar subgraphs in the input graph. Similarly, the interpretability of another framework called MICRO-Graph [143] is supported by the learned motif embedding and node-to-motif assignment.

Rationale Generation. In this type of method, the self-interpretable GNNs incorporate a module that can produce a subgraph rationale for the input graph data. For example, Yu *et al.* [144] propose a method called GIB, which is based on the information bottleneck theory. It can generate subgraphs that share the most similar property (i.e., label) to the input graphs. Adopting a causal view of graph generation, Wu *et al.* [126] propose a framework called DIR, which can produce rational subgraphs that are invariant to the changed data distribution.

2) Post-hoc Explainers for GNNs: Explainers for GNNs include gradient/feature-based methods, perturbation-based methods, surrogate methods, decomposition methods, and generation methods.

Gradient/Feature-based Methods. Gradient-based methods employ different types of gradient processing to obtain an estimation of the contribution made by components in the input samples. These methods produce local explanations for

current samples by calculating the gradients of GNNs on input space. Gradient-based methods treat input samples, rather than model parameters, as variables of GNNs, and utilise the backpropagation mechanism to obtain the gradients of GNNs on the input space. For example, a method called SA [145] uses the square value of gradients to define the contribution of components (e.g., nodes, edges or node features) in the current graph.

Feature-based methods use mappings from embedding space to input space to evaluate the contribution made by components in the input samples. For example, Pope *et al.* [58] propose an algorithm called CAM, which maps the embedding in the last layer to the input space by means of weighted summation. The mapping weights come from the last fully-connected layer used for the final prediction. The main differences between feature-based methods are the mapping construction methods employed. Moreover, according to a recent study [146], introducing regularisation terms (i.e., batch representation orthnormalisation and Gini regularisation [146]) can improve feature-based methods like CAM [58] and Grad-CAM [58].

Perturbation-based Methods. The counterfactual explanation describes causality as "If X had not occurred, Y would not have occurred" [130], [136]. In the context of explainers for GNNs, when removing one edge, node or changing one node feature from the explanation for the current output, GNNs will generate a different output if the removed or changed component is a counterfactual explanation for the current output. Accordingly, different from gradient/feature-based methods, perturbation-based methods attempt to find the counterfactual explanation [22] or underlying cause [147], [148], [149] for GNNs by perturbing components of the graphs. Specifically, perturbation-based methods apply masks to edges, nodes, or node features to filter out the most essential components, which serve as explanations for GNNs. It is expected that the predictions of GNNs on explanations will be similar to those on input samples. For example, GNNExplainer [22] utilises mutual information to train mask generators, which can provide explanations for both node classification and graph classification. Differences between perturbation-based methods stem primarily from their mask types, metrics for component importance, mask generation algorithms, and the forms taken by their explanations (e.g., edges, features or subgraphs).

Surrogate Methods. It is sometimes difficult to employ gradient/feature-based methods and permutation-based methods that require internal knowledge about GNNs (e.g., node embedding in PGExplainer [56]) because access to the GNN system may be limited (e.g., only queries are allowed). A practical solution is to employ interpretable and straightforward models (i.e., surrogates) to simulate the input-output mapping in target GNNs. By considering the similarity between the input samples and their neighbour samples, the functionally similar surrogate model can interpret the local behaviours of target GNNs to some extent. For example, a method called PGM-Explainer [150] perturbs node features to obtain similar samples, then employs an interpretable Bayesian network to provide explanations. Surrogate methods are generally model-agnostic; their differences mainly lie in the neighbour sampling strategies and interpretable surrogate models employed.

Decomposition Methods. Unlike the above methods, decomposition methods distribute the prediction score of GNNs on samples to the input space according to certain decomposition rules that help identify which components of the input space provide the greatest contribution to the prediction of GNNs. For example, the GNN-LRP [151] algorithm employs Taylor decomposition as rules to model interactions between GNN layers and then identifies a collection of walks to explain the output of GNNs. For decomposition methods, the decomposition rules are the main differences between them.

Generation Methods. In generation methods, explanations for GNNs are produced directly from the explanation generator module by employing graph generation methods based on reinforcement learning (RL) or other frameworks. For example, a method called XGNN [132] aims to obtain a graph pattern that maximises the prediction of GNNs on the target class, after which this graph pattern is used as an explanation for GNNs. This approach formulates graph generation as a reinforcement learning task. In each step of graph generation, XGNN makes an edge-adding decision based on the existing graph pattern. It uses policy gradients and feedback from GNNs to train the graph pattern generator. In comparison to other methods, the class-level explanation obtained from XGNN provides a high-level understanding of GNNs and a human-intelligible subgraph explanation of GNNs for graph classification. In addition to using reinforcement learning frameworks (e.g., XGNN [132], RG-Explainer [152]), other graph generation frameworks can also be employed to provide GNN explanations. For example, adopting a causal theory perspective, Lin *et al.* propose a method called OrphicX [153] based on the variational graph auto-encoder (VGAE).

Others. This category contains methods that cannot be categorised into the above post-hoc explanation methods. For example, Feng *et al.* [154] propose a method called DEGREE to estimate the contribution score of nodes and employ an agglomeration algorithm to complete the explanation subgraph construction. In DEGREE, based on the forward operation decomposition on GNN layers, the node contribution is calculated as the relative contribution (like the Shapely value [130]) when adding the target node to its contexts. Abrate and Bonchi [127] propose a heuristic-search method called OFS+OBS that finds counterfactual graphs to explain graph classification results on brain networks. Lin *et al.* [155] propose a framework called Gem to train an explanation generator; under this approach, the ground-truth explanations in training data are distilled through combining predefined rules and queries on target GNNs.

In addition to the above methods, some model-agnostic methods for general AI can also be extended to explain GNNs. For example, IG [156], DeepLIFT [157] and CXPlain [158] can be transferred to the graph domain to provide explanations for graph classification [149].

Remarks. After obtaining explanations of GNNs via the above methods, it is possible to evaluate the quality of these explanations through visualisation and accuracy-related metrics (see Section I-C). In Appendix A, we present related resources for

TABLE IV
COMPREHENSIVE COMPARISON OF TYPICAL METHODS FOR IMPROVING THE INTERPRETABILITY AND EXPLAINABILITY OF GNNS.

Methods	Category	Methodology	Explanation			
			Knowledge*	Level	Task*	Form
NGF [139]	Interpretability	Contribution	White-box	Instance	GC	Node
SAN [133]	Interpretability	Contribution	White-box	Instance	GC	Edge
STANE [140]	Interpretability	Contribution	White-box	Instance	LP/NC	—
SE-GNN [141]	Interpretability	Interpretable Module	White-box	Instance	NC	Subgraph
PortGNN [142]	Interpretability	Prototype	White-box	Instance	NC/GC	Subgraph
MICRO-Graph [143]	Interpretability	Prototype	White-box	Instance	GC	Subgraph
DIR [126]	Interpretability	Rational	White-box	Instance	GC	Node/Edge
GIB [144]	Interpretability	Rational	White-box	Instance	GC	Subgraph
SA [145]	Explainability	Gradient/Feature-based	Grey-box	Instance	NC	Node/Edge
Guided BP [145]	Explainability	Gradient/Feature-based	Grey-box	Instance	NC	Node/Edge
CAM [58]	Explainability	Gradient/Feature-based	White-box	Instance	GC	Node
Grad-CAM [58]	Explainability	Gradient/Feature-based	White-box	Instance	GC	Node
GNNExplainer [22]	Explainability	Perturbation-based	Grey-box	Instance/Group	NC/GC	Edge/Feature
PGExplainer [56]	Explainability	Perturbation-based	Grey-box	Instance	NC/GC	Edge
ZORRO [159]	Explainability	Perturbation-based	Grey-box	Instance	NC	Node/Feature
Causal Screening [149]	Explainability	Perturbation-based	Grey-box	Instance	GC	Edge
GraphMask [160]	Explainability	Perturbation-based	White-box	Instance	SRL/MQA	Edge
SubgraphX [161]	Explainability	Perturbation-based	Black-box	Instance	NC/GC	Subgraph
CF-GNNExplainer [162]	Explainability	Perturbation-based	Grey-box	Instance	NC	Edge
RCExplainer [136]	Explainability	Perturbation-based	Grey-box	Instance	NC/GC	Edge
ReFine [163]	Explainability	Perturbation-based	Grey-box	Instance	GC	Edge
CF^2 [164]	Explainability	Perturbation-based	Grey-box	Instance	NC/GC	Edge/Feature
GraphLime [135]	Explainability	Surrogate	Black-box	Instance	NC	Feature
RelEx [165]	Explainability	Surrogate	Black-box	Instance	NC	Edge
PGM-Explainer [150]	Explainability	Surrogate	Black-box	Instance	NC/GC	Node
LRP [145]	Explainability	Decomposition	White-box	Instance	NC/GC	Node/Edge
Excitation EB [58]	Explainability	Decomposition	White-box	Instance	GC	Node
GNN-LRP [151]	Explainability	Decomposition	White-box	Instance	GC	Walk
XGNN [132]	Explainability	Generation	Black-box	Class	GC	Subgraph
RG-Explainer [152]	Explainability	Generation	Black-box	Instance	NC/GC	Subgraph
OrphicX [153]	Explainability	Generation	Grey-box	Instance	NC/GC	Subgraph
DEGREE [154]	Explainability	Others	White-box	Instance	NC/GC	Subgraph
OFS+OBS [127]	Explainability	Others	Black-box	Instance/Class	GC	Node/Edge
Gem [155]	Explainability	Others	Black-box	Instance	NC/GC	Edge/Subgraph

* In the Knowledge column, "White/Grey/Black-box" indicate that the target GNN is treated as a white/grey/black box when obtaining explanations on it. The "GC", "LP", "NC", "SEL" and "MQA" in the Task column represent graph classification, link prediction, node classification, semantic role labelling, and multi-hop question answering, respectively.

studying the explainability of GNNs.

C. Summary

Table IV presents a comprehensive comparison of representative methods that provide explanations for GNNs. In the below, we also briefly compare them from several high-level perspectives.

1) Interpretability and Explainability: Interpretability and explainability refer to self-interpretable GNNs and post-hoc explainers for GNNs, respectively. Generally speaking, self-interpretable GNNs can only provide explanations for themselves, since the explanations are dependent on their specific architectures. Compared with most self-interpretable GNNs, post-hoc explainers have a broader range of applications, because they can usually provide explanations for GNNs with any architecture. However, the above difference does not mean that the functionality of self-interpretable GNNs is limited. For example, one recent GIB framework [144] simultaneously demonstrates the capability to discover the most informational subgraphs and flexibility (i.e., plug-and-play) in cooperating with other GNN backbones.

2) White/Grey/Black-box Knowledge: When obtaining explanations from self-interpretable GNNs, users typically require to have white-box knowledge on target GNNs. Here, we compare these existing post-hoc explainers based on the levels of background knowledge required.

Gradient/feature-based methods explicate GNN predictions in a straightforward manner. However, these methods usually require access to specific knowledge (e.g., inner gradients or features) about target GNNs (i.e., they treat the target GNNs as white/grey boxes). Perturbation-based methods generally adopt a causal theory-based approach, and accordingly need to learn different kinds of masks to generate explanations. Most of them also require access to specific knowledge (e.g., backward gradients) about target GNNs to learn how to generate these masks. Compared with other methodologies, surrogate methods are more practical, since they only need to

query target GNNs (i.e., they treat GNNs as black boxes). In decomposition methods, explanations are derived by distributing final predictions into input space. Thus, these methods also need access to knowledge about GNNs (i.e., white-box setting) to implement the above distribution. For their part, generation methods can directly produce human-intelligible subgraph explanations for specific input graph data or target classes. Most of these approaches treat target GNNs as black boxes when training the explanation generator.

3) Reasoning Rationale: From the causal perspective, most of today's explanation methods are based on factual reasoning (e.g., GNNExplainer [22], PGExplainer [56] XGNN [132], RG-Explainer [152], OrphicX [153]) or counterfactual reasoning (e.g., CF-GNNExplainer [162], Gem [155]). One recent study [164] shows that considering only factual reasoning will result in extra information being included in explanations (i.e., sufficient but not necessary explanations), while only considering counterfactual reasoning will break the complement graph of explanations (i.e., necessary but not sufficient explanations). Existing approaches that consider both forms of reasoning (e.g., RCExplainer [136], CF^2 [164]) show superiority in terms of explanation robustness [136] and quality (e.g., accuracy, precision) [164].

4) Other Limitations: In these gradient-based methods, contribution values based on gradients may only reflect the local sensitivity of GNNs; thus these methods may suffer from saturation problems [157] (i.e., the gradient of model output w.r.t the inputs is zero) and explanation misleading [166], [136]. In perturbation-based methods, the continuous values in soft mask learning methods (e.g., GNNExplainer [22]) may suffer from the "introduced evidence" problem [167] (i.e., potential side effects utilised by models when introducing mask operations) [29]. Although discrete masks in some methods (e.g., ZORRO [159] Causal Screening [149]) can alleviate this issue, the explanation generation mechanism (e.g., greedy algorithms) employed therein may result in locally optimal explanations. For surrogate methods, there is no consensus on how to define neighbours of the input graph data and how to choose interpretable surrogate models [29]. Moreover, the white-box knowledge setting of decomposition methods may make them impractical for users who can only query GNNs (e.g., using cloud-based GNN services [168]). Finally, a potential weakness of existing generation methods is that they ignore counterfactual explanations (i.e., they only consider the relationship between explanations and the labels of input graph data in their reward or loss functions).

D. Future Directions of Explainable GNNs

Strictly Model-agnostic Methods. Compared with self-interpretable GNNs, one advantage of post-hoc explainers for GNNs is their ability to explain GNNs with various architectures. However, only a few existing explainers are strictly model-agnostic (i.e., black-box in Table IV) methods (see Section IV-C). Of the existing explainers, the methods that require black-box knowledge of GNNs (see Section IV-C) present stronger practicality, as users sometimes cannot obtain enough knowledge about GNN systems to support a white-box approach. For example, when using cloud-based GNN services [168], users can only query GNN services to obtain predictions on the input graph data, and cannot access the inner backward gradients of GNNs, which are indispensable for the training of some explainers (e.g., GNNExplainer [22] and PGExplainer [56]). Due to the diverse application scenarios of post-hoc explainers, designing compelling and low-demand explainers represents a promising avenue for facilitating the explainability of practical GNN systems.

Evaluation Benchmark for Real Applications. The absence of real-world ground truth datasets hampers the exploration and identification of practical GNN explanation methods. Existing explanation methods are generally evaluated on synthetic datasets with visualisation and accuracy-related metrics [29]. Although some methods, like GNNExplainer, obtain competent performance on these datasets [22], they have limited explanation accuracy for GNNs on real-world datasets such as scene graphs [149]. Therefore, designing evaluation datasets and metrics for different applications is crucial for boosting GNN explanation methods and comprehensively evaluating their performance. Another possible solution is to explore the integration of visual analytics and interpretable GNNs, which can facilitate explanation evaluation in real-world applications.

V. Privacy of GNNs

Private GNNs require that confidential data (such as model parameters and graph data) should not be leaked. GNNs have been used in sensitive fields, and GNN users place a high priority on the protection of their personal information. A medical record system, for example, might contain a graph that represents the social connections among patients infected with COVID-19 [169]. It is important to safeguard patients' sensitive personal information while learning GNN models from these data.

In this section, we present recent studies that have examined privacy issues related to GNNs. First, we introduce some privacy-related attacks, which reveal the privacy risks in GNN systems. We next discuss several privacy-preserving techniques used in GNN systems. Finally, we provide an outlook of the future directions in GNN privacy.

A. Privacy Attacks

GNN systems face the threat of data privacy breaches. Through the exploitation of certain vulnerabilities, several privacy attacks have been proposed with the goal of inferring sensitive information from GNN systems. Specifically, there are two main factors to consider when analysing these attacks: attack targets and attack knowledge.

Attack Targets. Attackers attempt to steal the private information contained in GNNs, including models and graph data.

- *Models.* GNN models, including both model architectures and parameters, often represent the intellectual properties of model owners. For example, an e-commerce company may employ a confidential GNN for recommendation [170], [171], making it an important commercial property, and revealing model will have grave privacy implications.

- *Graph data.* Graph data, which also contains a high degree of sensitive information, should be kept private. For example, patients expect to use GNNs for medical diagnosis without their personal medical profiles being leaked [172], [173]. Note that, in the graph data context, "privacy" refers to not only the privacy of the graph samples, but also the graph statistical properties and training membership (i.e., whether or not a node/edge/graph sample is engaged in GNN training).

Attackers' Background Knowledge. Attackers are assumed to have access to different knowledge about target GNNs.

- *Black-box knowledge.* In black-box settings, attackers can only send queries to GNN models, but cannot access the training graphs and inner knowledge of GNN models [174], [45]. For example, in the context of Machine Learning as Service, attackers have a similar status to end-users in that they only have access to the public APIs of the models.
- *White-box knowledge.* Attackers can also obtain white-box access to the target GNN models in certain scenarios [174], [175]. Note that, unlike the setting of adversarial attacks in Section III-A1 (where the entire training process is known), white-box attackers in this context can only access GNN models, while the training graphs and labels remain unknown. This scenario often arises when model owners publish their GNNs while still wishing to keep their training data and training strategies secret.

In this section, we summarise existing privacy attacks and introduce them based on the above-mentioned aspects.

Model Extraction Attacks. Model extraction attacks are designed to steal information about the architecture and parameters of a GNN model. Specifically, they can reconstruct the original model, or construct a substitute model that performs similarly to the original. Current model extraction attacks on GNNs focus only on node classification tasks [60]; in comparison, there is little research on attacks against graph classification and link prediction. The attack methods are designed based on different background knowledge of the attackers. The attack methods are designed based on different background knowledge of the attackers. For example, knowing the node attributes of a set of adversarial nodes enables attackers to use discrete graph structure learning methods (e.g., LDS [176]) to construct a connected substitute graph based on these node attributes. The attackers can then generate queries from these adversarial nodes and use the responses to train a substitute GNN [60].

Membership Inference Attacks. Membership inference attacks aim to infer whether a specific component or sample has been included in the training dataset of a victim GNN. These attacks can be categorised depending on the GNN target task involved.

- *Node-level attacks.* In node-level classification tasks, membership inference attacks aim to determine the membership of a specific node. Specifically, attackers intend to determine whether a node has been used as a training node for GNNs [174], [46].
- *Link-level attacks.* As representations of nodes' relationships, edges also contain private information and have thus become common targets of membership inference attacks. This type of attack attempts to determine whether a specific link between two nodes exists in the training graph [45]. Since GNNs essentially aggregate information about each node from its neighbours, the output posteriors of two connected nodes are likely to be closer together than those of non-connected nodes. Therefore, using only the posteriors of nodes obtained from the target model, the attackers can calculate the distance between posteriors of a node pair and select a threshold to determine the membership of the links.
- *Graph-level attacks.* Graph-level membership inference attacks are designed to identify the membership of an entire graph rather than the individual components (i.e. a single node or edge) within it. Recent work [47], [177] has demonstrated that graph-level GNNs with several popular architectures are vulnerable to membership inference attacks.

Model Inversion Attacks. Model inversion attacks aim to extract information about model inputs from their corresponding outputs. Specifically, attackers might exploit a GNN's outputs (e.g. node embedding and graph embedding) to obtain sensitive information about input graphs (e.g., node attributes or graph properties). Two popular types of inversion attacks are introduced here based on the information they attempt to recover:

- *Property inference.* Such attacks attempt to infer basic properties of the target graph (i.e. the number of nodes, edges, and graph density) from the graph embedding [175].
- *Graph reconstruction.* This type of attack aims to directly reconstruct the original graph or attributes based on the output embedding. For graph structure reconstruction, given a targeted GNN and some other knowledge (such as node labels and attributes), attackers can recover a specific subgraph or the entire graph [174], [178].

Other Privacy Attacks. In addition to the above attacks, researchers also exploit privacy leakages from other attack surfaces. For example, considering GNN systems with a vertical input graph partition (e.g., node features and edges are held by different individuals), attackers who know the node features can infer the private edges held by other partitions [179]. Another type of attack involves inferring the node labels from a link prediction GNN, in which node labels are supposed to be hidden from users [61].

Summary. The attacks discussed above consider different targets, and thus have different design rationales. Model stealing and graph property inference are primarily based on the strong relationship between the query input, the output, and the learned GNN models. The model extraction attack, for example, reconstructs a model based on the mapping between input and output. Property inference is also based on this mapping, but for attribute inference. Alternatively, membership inference attacks make use of the fact that the GNN model remembers the training data, commonly identifying the difference between members and non-members based on the output posteriors alone.

Unlike privacy attacks on DNNs, these privacy attacks on GNNs may benefit from the unique properties of graphs and

GNNs. As an example, the effectiveness of stealing links [45] (membership inference attacks targeting an edge in the training graph) exploits the similarity between the outputs of two connected nodes. Moreover, a model extraction attack [60] simulates the propagation of information in GNNs in order to construct a surrogate graph.

B. Privacy-preserving Techniques for GNNs

In light of the privacy threats associated with GNN systems, several privacy-enhancing techniques for these systems have been developed. In this section, we will introduce several of the most popular approaches.

1) Federated Learning: Federated Learning (FL) [180] is a popular paradigm that enables individuals (e.g., mobile devices) to train ML models collaboratively without revealing each individual's raw data. In FL, the data belonging to these decentralised individuals can be considered private, so it is processed and trained locally without being shared with others. these decentralised individuals' data are considered private, processed, and trained locally without sharing to others. To facilitate privacy-preserving training, a server continuously gathers and aggregates parameters (e.g., gradient/model weights) from each individual until the model performance converges. Recently, FL has also been increasingly investigated and applied in the GNN context. Table V summarises several existing studies about FL in GNNs. In particular, based on how the graph is distributed to individuals, we introduce two types of FL in the GNN context:

Inter-graph FL. Inter-graph FL [181] can be considered as a natural extension of DNN-based FL tasks (e.g., FL for image classification). For this particular type, each individual processes its own set of local graph samples, performs training collaboratively, and participates in producing a global GNN coordinated by a central server. A typical application of inter-graph FL can be found in the biochemical industry, where each pharmaceutical company possesses a confidential dataset consisting of a collection of molecule graphs. These companies perform inter-graph FL to train a global GNN for drug property analysis without the need to pool their confidential data into a central server [182].

Intra-graph FL. Intra-graph FL is designed for scenarios in which multiple clients want to jointly train a GNN on a global graph, with each of them owning a local subgraph of this global graph [181], [183]. Based on the distribution characteristics of the global graph (i.e., how the entire graph is distributed to each client in the feature and node space), there are two types of intra-graph FL [184]:

- *Horizontal intra-graph FL.* This type considers the case in which the local (sub)graphs belonging to each individual are horizontally partitioned. In this case, a global graph is dispersed over multiple individuals, each of whom owns partial nodes and is interested in training collaboratively without data leakages [31], [185]. For example, in social networking apps, each user has a local social network, which is a subgraph of the overall human social network [186]. Through the use of horizontal inter-graph FL, a global GNN for recommendation can be generated while protecting the private graph data of each of these instances.

TABLE V
A COMPARISON OF TYPICAL METHODS FOR THE FEDERATED LEARNING OF GNNS.

Method	Category	Task
Feddy [191]	Inter	GC
SpreadGNN [192]	Inter	GC
GCFL [193]	Inter	GC
FedCBT [194]	Inter	GC
FedChem [195]	Inter	GC
STFL [196]	Inter	GC
FedGNN [197]	Inter	R
FeSoG [198]	Inter	R
FedVGCN [199]	Intra(V)	NC
VFGNN [200]	Intra(V)	NC
ASFGNN [186]	Intra(H)	NC
GraphFL [185]	Intra(H)	NC
FedGL [201]	Intra(H)	NC
CNFGNN [202]	Intra(H)	NC
FedSage/FedSage+ [203]	Intra(H)	NC
FedGraph [204]	Intra(H)	NC
SAPGNN [205]	Intra(H)	NC
FedGraphNN [31]	Inter/intra(H)	GC/NC

* "Inter" indicates Inter-graph FL. The "Intra(H)" and "Intra(V)" indicate Horizontal intra-graph FL and Vertical intra-graph FL, respectively. The "NC", "GC", and "R" stand for Node Classification, Graph Classification and Recommendation, respectively.

- *Vertical intra-graph FL.* This type of FL considers a scenario in which the subgraphs owned by each individual are vertically partitioned. Specifically, the subgraphs of individuals are distributed in feature space; while each of them possesses a different feature and label space, they all share the same nodes and their connections. The FL method is commonly used to train global GNNs for multiple cooperative organisations. For instance, to develop a graph-based financial fraud detection system [187], banks and regulators with common customers (e.g., Webank [188] and the National VAT Invoice Verification Platform [189]) opted to train a GNN collaboratively; these companies share the same nodes (common customers), but each have their own distinct feature space (loan records in different institutions) [190].

Similarly to FL in DNNs, FL in GNNs protects the private data of each individual by transmitting model parameters rather than raw data between the individuals and central server during training. In this way, private data can be kept local and cannot be accessed by others. Specifically, each individual can train a GNN model with its own data. Subsequently, they can upload their local model to the server, which then performs an aggregation function (such as FedAvg [26]) to build a global GNN model. Since such aggregation uses GNN parameters inductively (i.e., the model is independent of its structure), topological information about the local graph data is not required and can accordingly be kept private.

Unlike DNN-based FL tasks, FL in GNNs often incorporates carefully designed mechanisms. Due to the characteristics of graph data, certain challenges that arise during the training or implementation of FL systems may manifest or be amplified in the GNN context. These challenges can be summarised as follows:

- *Non-IID individual data.* Similar to ordinary DNNs, FL in GNNs also relies on stochastic gradient descent (SGD), which makes their training performance easily be affected by non-IID training data [206]. In practice, due to the heterogeneity of the structures and features of the graphs owned by different individuals, non-IID individual data (i.e., diverging graph structure distributions and node feature distributions of the local graph data) is an inevitable problem for FL with graphs [193]. Thus, most FL methods desgined for GNNs aim to to minimise the impact of non-IID data on the training performance. As an example, ASFGNN [186] decouples the FL training and refines the global model parameters based on the JS-divergence to ensure that the loss computation is not biased by the individual data. In GraphFL [185], a meta-learning technique called model-agnostic meta-learning is employed, which has been proven to deal efficiently with non-IID data.
- *Graph isolation.* In horizontal intra-graph FL, local graph data can be considered as subgraphs isolated from a latent global graph. Since representation learning on graph models relies on messages passing through the connections, private information from the other subgraphs cannot be gathered, which will impact the accuracy of the GNNs. Existing FL methods designed for GNNs address this problem by transmitting the higher-domain representation rather than the raw connections [203], [204] or by using a generator to generate missing neighbours across distributed subgraphs [202].

2) Differential Privacy: Differential privacy (DP), as a well-known technique for privacy-preserving ML algorithms, can guarantee that an attacker will be unable to derive private information regarding a specific piece of training data from a released learning model, with high confidence [207]. Providing such a guarantee of privacy is also a crucial requirement when developing privacy-preserving GNN applications. As an example, a social smartphone application server stores information on social interactions between its users. The server may, however, also want to utilise users' private data, such as lists of installed applications or usage logs, in order to develop better GNN models and provide better services (e.g., recommendation system) [208]. Due to privacy concerns, the server should not access users' raw data without any data protection.

To solve this problem, the key idea behind DP is that rather than disclosing their private data, data owners are advised to add noise to their data before sending them to the server. The data are perturbed in such a way that they will appear meaningless when viewed individually, but approximate the analytics result when aggregated [208]. In prior studies of applying DP to machine learning algorithms using SGD [207], [209], perturbation is added to the gradients generated during the model training. In GNNs, the perturbations can be applied to node features [208], [210]. For example, consider a case in which DP is used in a node classification training task [208]. In this task, a server aims to learn a GCN model that can classify nodes without knowing the private node features of the users. Users can add noise to their private feature $\mathbf{X}_{i,d} \in \mathbb{R}$, which denotes a feature value of the d-th dimension in $\mathbf{X}_i$ for node v_i, so that the perturbed features $\mathbf{X'}_{i,d} \in \mathbb{R}$ satisfy the following conditions:

$$\mathbb{E}[\mathbf{X'}_{i,d} - \mathbf{X}_{i,d}] = 0.$$

When the server performs a mean aggregation (can also be extended to other aggregation functions), for any node $v \in \mathcal{V}$ and any feature dimension $d \in \{1, 2, ..., \mathcal{D}\}$:

$$\begin{cases} \mathbf{M}_{v,d} = \dfrac{1}{\mathcal{N}(v)} \displaystyle\sum_{u \in \mathcal{N}(v)} \mathbf{X}_{u,d} \\ \mathbf{M}'_{v,d} = \dfrac{1}{\mathcal{N}(v)} \displaystyle\sum_{u \in \mathcal{N}(v)} \mathbf{X}'_{u,d} \end{cases}$$

Thus, based on Bernstein inequalities, it satisfies the following:

$$\Pr\left(\|\mathbf{M}'_{v,d} - \mathbf{M}_{v,d}\| \geq \lambda\right) \leq \epsilon,$$

where $\Pr$ is the probability, λ and ϵ are hyper-parameters used for adjusting the utility and privacy. As a result, the server can use these approximations of aggregation results generated from perturbed inputs and achieve privacy preservation.

3) Insusceptible Training: Some other studies [62], [211], [212] have attempted to defend against privacy attacks and reduce the leakage of sensitive information via modifying the training process of GNNs. For example, it is possible to add privacy-preserving regulation items in the loss function during GNN training, or introduce privacy-preserving modules in GNN architectures to reduce privacy leakage [62]. Specifically, consider a defender aiming to defend against a private attribute inference attack $\mathcal{F}_A(v_i)$. The defender can modify the original objective functions by adding a privacy leakage loss during the target GNN training, as follows:

$$\min_{\theta} \sum_{v_i \in \mathcal{V}} \mathcal{L}_Y(f_\theta(v_i)) + \lambda \mathcal{L}_A(\mathcal{F}_A(v_i)),$$

where $\mathcal{L}_Y(\cdot)$ and $\mathcal{L}_A(\cdot)$ are the loss function of the target GNN utility and attribute inference attack, respectively. Other types of defences propose to filter out the sensitive attributes by learning GNN encoders [211], [212].

4) Security Computation: Security computation has also been widely adopted to protect data privacy in general data analytics systems and services. There are three main types of techniques for security computation, including Trusted Executive Environment (TEE) [213], [214], Homomorphic Encryption (HE) [215], [216], and Multi-party Secure Computation (MPC) [217], [218], [219]. Note that, since the underlying operations (e.g., matrix multiplication) in GNNs are similar to those in ordinary DNNs, most of the existing security computation methods in DNNs can be directly extended to GNNs.

5) Summary: We examine and compare the above privacy-preserving methods from two perspectives:

Application Scenarios. Insusceptible training is designed to protect one type of sensitive information (e.g., a specific sensitive attribute). As a consequence, it cannot protect against other privacy threats that target other types of private information [211], [212]. Meanwhile, FL seeks to protect local data in a distributed learning system [31], [191]. Local users will always hold their personal data, meaning that the privacy of

all their local data (rather than certain attributes only) can be guaranteed. Moreover, FL is designed for collaborative model training, meaning that it cannot directly protect against attacks that occur during the inference process (e.g., model extraction attacks and membership inference attacks) [211]; by contrast, insusceptible training can prevent the leakage of private information during inference periods.

Trade-off between Privacy, Accuracy, and Efficiency. As previous studies have shown, there is a trade-off between privacy, accuracy, and efficiency [207], [208]. For example, the introduction of DP into a GNN will reduce its overall accuracy. One recent study using DP examined this trade-off between privacy and accuracy [208], [210]. On the other hand, these privacy preserving-techniques also lead to a higher time expenditure. In the context of federated learning, multiple rounds of communication among different individuals greatly increase the time cost of completing the model training [220], [180]. Moreover a more critical bottleneck is the increase in latency during inference when using security computation techniques. In the case of privacy-preserving inference using HE and MPC, the inference latency will increase significantly, becoming orders of magnitude higher than the latency of computation in cleartext [215], [221].

C. Future Directions of Private GNNs

Leakage from Gradient. The risks caused by leakage from gradient [222] have not been fully explored in the GNN context. Specifically, most current works focus on discussing the potential risks caused by the leakage of final prediction results (such as prediction labels [60], confidence scores [45], etc.). However, final results are not the only information that can give rise to privacy leakage; an attacker can also infer sensitive information from other types of knowledge. It was recently determined that leaking gradients containing training or inference information can also lead to serious privacy issues [222]. In the case of FL, for example, attackers can reconstruct the private training data of local clients through the sharing gradient, which is also known as a gradient inversion attack [223], [224]. Further privacy analysis on how leakages of gradients can harm GNN privacy should be thoroughly explored.

Defence against Privacy Attacks. Despite the fact that several privacy-preserving techniques are available for GNNs, not all of them are suitable for defending against privacy attacks. Specifically, many privacy attacks operate under black-box settings in which attackers can only send black-box queries to target GNNs [174], [45]; in short, they can infer sensitive information without knowing the internal operations of a target GNN system. It should be noted that many privacy-preserving GNN techniques keep graphs or GNN models secret during the process of training or inference [191], [186]; they are not designed to prevent the information leakages revealed by the intended results (e.g., trained GNN models or prediction results). Accordingly, they do not provide formal privacy guarantees with regard to the indirect leaks of information when an attacker attempts to infer sensitive information through privacy attacks. As an example, federated learning on GNNs offers a method for protecting private data by storing it locally on the client side, and training GNNs based only on gradients uploaded by clients [184], [202]. After GNN training is complete, attackers can still infer membership of elements in the graph data from the final well-trained model [174], [45]. It is therefore necessary to further explore how to build a privacy-preserving GNN system that is able to defend against these privacy threats.

VI. Fairness of GNNs

Fair systems win people's trust by protecting the vital interests of vulnerable groups or individuals. In the context of GNNs, fairness means that prejudice or favouritism towards an individual or a group is excluded from GNN predictions. As the generalisation of deep neural networks on the graph domain, current GNNs are data-driven and designed to learn desired mappings from graph data. It should be noted here that elements of the GNN learning process, such as message-passing (see Equation 1), can amplify historical or potential bias in graph data, which results in discrimination or bias in predictions of GNNs with respect to sensitive attributes (e.g., skin colour) [21].

In this section, we summarise current efforts targeted at achieving fairness in GNNs. We first introduce definitions and the metrics commonly used to measure fairness. Next, we present the common ideas underpinning existing methods. Finally, we discuss future directions in exploring fairness enhancements for GNNs.

A. Concepts and Categories

Bias, Discrimination, Fairness. Bias and discrimination are two common concepts that are related to unfairness in AI-related research; unfairness stems from both of these factors [41]. In the pipeline of AI systems, the main types of bias are *data to algorithm bias*, *algorithm to user bias*, and *user to data bias* [41]. *Bias* comes from unfair operations in data collection, sampling, and measurement, while *discrimination* is caused by intentional or unintentional human prejudices and stereotyping with regard to sensitive attributes (e.g., race) [41].

Fairness is an essential and indispensable factor in maintaining social order. The disciplines of philosophy and psychology have a long history of fighting against bias and discrimination, dating back to well before the birth of computer science [41]. However, owing to the diversity of cultures and application contexts involved, there is no universal definition of fairness. *Fairness* can be broadly defined as *"the absence of any prejudice or favouritism towards an individual or a group based on their intrinsic or acquired traits in the context of decision-making"* [225], [41].

Unfairness Cycle. Bias emerges from biases in data, algorithms and user interactions [41]. As shown in Fig. 4, during the generation of graph data (e.g., annotation [10]), human behavioural bias or content production bias can in turn introduce bias into graph data. For example, in Pokec (a popular social network in Slovakia), the number of intra-group edges far exceeds the number of inter-group edges [21]. During the learning of GNNs, due to the operation bias

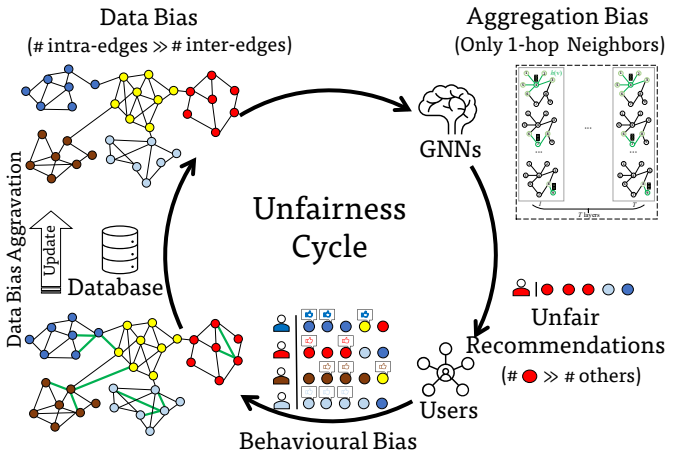

Fig. 4. The Unfairness Cycle in Systems with GNNs. The bias circulates between graph data, graph neural networks and users.

(i.e., the aggregation that only considers 1-hop neighbours) based on graph structures, the message-passing mechanism (see Equation 1) in GNNs can enlarge and perpetuate existing data bias by stacking graph filtering operations [21]. Moreover, some algorithmic operations (e.g., random walks) can also introduce unfair behaviours into the learning process [226]. Upon completion of deployments, these GNNs offer unfair recommendations to users. Subsequently, after receiving recommendations, users will interact with them according to their personal preferences. The behavioural biases of different users will further aggravate the bias in graph data.

Group Fairness, Individual Fairness, Counterfactual Fairness. In the context of GNNs, the three types of fairness most commonly discussed are group fairness, individual fairness and counterfactual fairness. *Group fairness* requires that different groups are treated equally by GNNs. A typical definition is demographic parity [227], which requires that the predictions $\hat{\mathrm{Y}}$ of GNNs satisfy

$$\Pr(\hat{\mathrm{Y}} \mid S = 0) = \Pr(\hat{\mathrm{Y}} \mid S = 1),$$

where $\Pr$ is the probability, and S indicates whether the sample comes from the protected (e.g., female) group [228]. Other common definitions of group fairness include conditional statistical parity, equalised odds, equal opportunity, treatment equality, and test fairness [41].

Individual fairness requires that similar individuals obtain similar predictions from GNNs. A typical description is the (d_1, d_2)-Lipschitz property [229], [33]:

$$d_1(f(x), f(y)) \leq L d_2(x, y) \quad \forall (x, y),$$

where $f(x)$ is the output of GNNs on sample x, L is a constant scalar, and $d_1(\cdot, \cdot)$ and $d_2(\cdot, \cdot)$ are similarity functions for the output and input space of GNNs, respectively. The (d_1, d_2)-Lipschitz property can also be used to measure the stability of $f(\cdot)$ when faced with slight perturbations in input samples.

Counterfactual fairness requires GNNs to make the same predictions for "different versions" of the same candidate [230]. It can be formalised as follows:

$$\Pr(\hat{\mathrm{Y}} \mid s = s_1, x = \mathbf{x}) = \Pr(\hat{\mathrm{Y}} \mid s = s_2, x = \mathbf{x}),$$

where condition $s = s_1, x = \mathbf{x}$ indicates version 1 of $\mathbf{x}$ with sensitive attribute s_1, and condition $s = s_2, x = \mathbf{x}$ indicates version 2 of $\mathbf{x}$ with sensitive attribute s_2.

Processing Stages. Based on the stage in which the methods are engaged, methods for achieving fairness can be categorised as pre-processing, in-processing and post-processing methods [41]. *Pre-processing* methods (e.g., debiasing the input graph in InFoRM [33]) attempt to remove the potential bias in the training data. *In-processing* methods (e.g., REDRESS [66]) modify the operations or objective functions of algorithms in training to avoid unfairness. *Post-processing* methods (e.g., debiasing results in InFoRM [33]) transform the outputs of models to achieve fairness in the results.

Explainable and Unexplainable Discrimination. Discrimination can be subdivided into explainable discrimination and unexplainable discrimination. If discrimination is *explainable*, this means that "differences in treatment and outcomes among different groups can be justified and explained via some attributes in some cases" [41]; thus, explainable discrimination is considered to be legal, reasonable and acceptable. For example, although the annual average income of females in the UCI Adult dataset is less than that of males [231], this is acceptable, since males work more hours per week on average than females. On the other hand, *unexplainable discrimination* is unacceptable, since it can not be justified with reference to acceptable reasons. The above discrimination categories commonly used in machine learning help to clarify the research goals of GNN fairness. In this survey, we focus on the latter kind of discrimination and the methods employed to alleviate it.

Unexplainable discrimination can be further subdivided into *direct discrimination* and *indirect discrimination* [41]. Direct discrimination occurs when some sensitive and protected attributes cause the undesirable results. For example, race, skin colour, national origin, religion, sex, familial status, disability, marital status, recipient of public assistance, and age are all protected attributes under the Fair Housing and Equal Credit Opportunity Acts [232]. Moreover, indirect discrimination refers to instances in which individuals are treated unjustly as a result of certain seemingly neutral and non-protected attributes in some sensitive applications. For example, the residential zip code, which may be related to race [233], can be employed as an essential factor in the loan decision making process.

Productive Bias, Erroneous Bias, Discriminatory Bias. Productive, erroneous, and discriminatory bias are concepts that have been proposed to add some nuance to the much-abused "bias" [10]. *Productive bias* indicates that an algorithm is biased towards certain distributions or functions, on which its performance is better. This conforms to the "no free lunch theory" [234]. *Erroneous bias* is the systematic error caused by faulty assumptions [10]. It indicates that the undesirable outputs of algorithms are caused by some biased operations (e.g., sampling bias), which cause the model to deviate from the assumptions (e.g., distribution consistency between training data and real data). Finally, *discriminatory bias* is used to describe an algorithm's unfair behaviours targeting a certain group or individual [10].

GNN fairness can be further clarified with reference to the

above bias categories used in AI. Productive bias is beneficial and acceptable, like *explainable discrimination*. Erroneous bias generally goes unnoticed by humans, since the violation of assumptions is difficult to measure. In this survey, we focus on the methods for solving unfairness under the scope of discriminatory bias, which, like *unexplainable discrimination*, is unjustified and unacceptable.

B. Methods

Current methods that aim to achieve fairness in the GNN context can be categorised into fair representation learning methods and fair prediction enhancement methods.

Fair representation learning is a typical method for achieving fairness in GNNs. In fair representation learning, sensitive information (e.g., gender) is excluded from learned representation to ensure the fairness of GNN predictions in downstream tasks. Within GNN architectures, adversarial learning is employed to achieve fair representation, and the goal of adversary modules is to infer sensitive attributes from learned representation. Fair representation is achieved once adversary modules are unable to infer these attributes. Two typical instances of fair representation learning are Compositional Filters (CF) [235], and FairGNN [21].

Fair prediction enhancement methods aim to achieve fairness in GNNs by introducing extra processes or operations into the GNN pipeline. Depending on the type of fairness involved, these methods can be categorised as group fairness, individual fairness, or counterfactual fairness enhancement methods. Current efforts utilise different strategies to eliminate bias in GNNs and thereby achieve fair predictions. These strategies include data augmentation methods [64], fair graph methods [32], and regularisation methods [33].

Under the data augmentation strategy, the counterfactual fairness on sensitive attributes can be regarded as invariant to the sensitive attribute values. Similar to adversarial training for robustness, the data augmentation strategy obtains "adversarial samples" by perturbing these protected attributes, then adds them into the GNN training data to achieve "robust" (i.e., fair) predictions on nodes. One representative example of data augmentation is a method called NIFTY [64]. The fairness in NIFTY is defined as counterfactual fairness, in which the encoder ENC satisfies the following:

$$\mathrm{ENC}(u) = \mathrm{ENC}(\tilde{u}^s),$$

where $\tilde{u}^s$ is obtained by modifying or flipping the sensitive feature s of node u. To achieve fairness in GNNs, NIFTY utilises the Siamese learning [236] approach and the augmented view of attribute information.

Fair graph methods aim to implement fairness-oriented modifications on the graph structure or node features before or during the training of GNNs. From the perspective adopted by these methods, biased predictions made by GNNs stem from unfair connections in the graph structure or biased node features. To remove bias from GNN predictions, these methods design different approaches including dropping edges that have been influenced by bias [65], learning new graph structures [33], [32], [237] or node features [237]. For link prediction tasks, Li *et al.* propose a method called FairAdj [32] and dyadic fairness, which is defined as follows:

$$\Pr(f(u,v) \mid S(u) = S(v)) = \Pr(f(u,v) \mid S(u) \neq S(v)),$$

where $f(u,v)$ is the link prediction function for node u and node v, and $S(\cdot)$ indicates the sensitive attribute value of nodes. FairAdj derives the bound for the discrepancy in link prediction expectations between inter-group and intra-group nodes. To minimise the upper bound, FairAdj is formulated as a bi-level optimisation that iteratively optimises the parameters of GNNs and fair connection to achieve fair link prediction. Another instance of fair graph structure methods is the method called FairDrop [65], which utilises edge-drop operations to reduce the unfairness impact caused by original graph structure.

Regularisation methods add extra regularisation terms to the objective functions or final predictions of GNNs to improve fairness. For example, a method called InFoRM [33] utilises the similarity matrix defined on nodes to obtain the regularisation term, which measures the individual fairness of GNNs. Notably, as nodes in a graph are connected by edges, it is not easy to achieve individual fairness on nodes by considering node features alone. InFoRM [33] employs a variant of the (d_1, d_2)-Lipschitz property to measure the individual fairness of nodes. It is defined as

$$\sum_{i=1}^{n}\sum_{j=1}^{n} \|\mathbf{Y}[i,:] - \mathbf{Y}[j,:]\|_F^2 \ \mathbf{S}[i,j] = 2\,\mathrm{Tr}\left(\mathbf{Y}'\mathbf{L}_{\mathrm{S}}\mathbf{Y}\right) \leq \delta,$$

where $\mathbf{Y}[i,:]$ and $\mathbf{Y}[j,:]$ are the graph mining results of node i and j, $\mathbf{S}[i,j]$ indicates the similarity of nodes, and $\delta > 0$ is a constant. The $\mathbf{L}_{\mathrm{S}}$ represents the Laplacian matrix of $\mathbf{S}$. The $\mathrm{Bias}(\mathbf{Y},\mathbf{S}) = \mathrm{Tr}\left(\mathbf{Y}'\mathbf{L}_{\mathrm{S}}\mathbf{Y}\right)$ is used to measure the overall bias in the result $\mathbf{Y}$ with respect to similarity $\mathbf{S}$. To achieve debiasing of GNNs, the item $\mathrm{Bias}(\mathbf{Y},\mathbf{S})$ is added to the loss function as a fairness regularization item during the training of GNNs. Moreover, this regularisation item can also be employed in the debiasing of input graph data and graph mining results. Thus, InFoRM can flexibly boost GNN fairness in any processing phase (pre-processing, in-processing or post-processing). Another typical example is the REDRESS [66] method, which defines a regularisation term from the ranking perspective.

Remarks. In Appendix A, we present further related resources for studying the fairness of GNNs. For fairness in general graph mining, the tutorial [230] introduces the definition of fairness in various tasks and the typical methods used to achieve it.

C. Summary

Table VI presents a comparison of the above-mentioned methods for improving the fairness of GNNs. Existing fairness enhancement methods on GNNs can be categorised into pre-processing, in-processing and post-processing methods. Pre-processing methods generally require modification of the training graph dataset (e.g., InFoRM-G [33]) of GNNs to remove potential discrimination in the data. In-processing methods

TABLE VI
A COMPARISON OF TYPICAL METHODS FOR IMPROVING THE FAIRNESS OF GNNS.

Methods	Fairness		Task	Phase
	Level	Goal		
CF [235]	-	Representation	LP	In
FairGNN [21]	-	Representation	NC	In
NIFTY [64]	C	Prediction	NC	Pre
EDITS [237]	G	Prediction	NC	Pre
InFoRM-G [33]	I	Prediction	LP/NC	Pre/In
FairAdj [32]	G	Prediction	LP	In
FairDrop [65]	G	Prediction	LP	In
InFoRM-M [33]	I	Prediction	LP/NC	In
InFoRM-R [33]	I	Prediction	LP/NC	Post
REDRESS [66]	I	Prediction	LP/NC	In

* The "G","M", and "R" in the InFoRM method indicate that the debiasing objects are the input graph, mining model, and mining results, respectively. The "G","I", and "C" in the "Level" column indicate group fairness, individual fairness, and counterfactual fairness, respectively. The "NC" and "LP" stand for node classification and link prediction, respectively. "Pre"/"In"/"Post" indicate the phases during which the method can engage.

modify the operation pipeline (e.g., learning fair graph structure in FairAdj [32], introducing an adversary module in FairGNN [21]) or loss function (e.g., REDRESS [66]) to improve fairness in GNN predictions. However, both pre-processing and in-processing methods are practical only when users are permitted to modify graph data and GNN models, respectively. If users can only treat GNNs as black boxes (e.g., when using cloud-based GNN services [168]), then users can only query GNNs and employ post-processing methods (e.g., InFoRM-R [33]) to alleviate predication unfairness in target GNNs.

D. Future Directions of GNN Fairness

Fairness Definition and Evaluation. The fairness of GNNs is an emerging topic in GNN research. The definition of fairness varies from method to method and is influenced by the task at hand. Some methods emphasise the group or individual fairness, while others focus on improving the independence between the learned representations and sensitive attributes [21], or the robustness of GNNs to the perturbation of sensitive attributes [64]. Thus, it is essential to arrive at a reasonable definition of fairness for various kinds of graph tasks (e.g., exploring dynamic graphs [238] or heterophilic graphs [239]) and applications is essential for building fairness in trustworthy GNNs. Moreover, the diverse range of definitions for the concept of fairness results in different metrics being employed for fairness evaluation. It is also necessary to build unified evaluation metrics and datasets.

Influence on Task Performance. The influence of promoting fairness on the original task performance of GNNs is unclear owing to complexity of graph data and the wide range of fairness definitions. On the one hand, introducing fairness to GNNs reduces GNNs' dependence on sensitive attributes. Existing research into algorithmic fairness [240] demonstrates that information reduction is harmful to model performance, resulting in the performance degradation of GNNs when considering fairness. On the other hand, a novel view of fairness may boost the performance of GNNs in some applications. For example, RawlsGCN [241] introduces the Rawlsian difference principle to GCN to alleviate the degree-related unfairness; results show that it can improve the performance of GCN on some datasets. To build trustworthy GNNs, it is necessary to explore the correlation between the fairness and performance of GNNs.

Revealing Unfairness. Although existing studies have demonstrated the unfairness of GNNs and proposed various methods to promote fairness, they cannot provide detailed explanations for the discrimination detected in predictions. One promising direction for fairness research is that of revealing the sources of unfairness, as these findings can subsequently guide the design of GNN architectures concerning fairness. For example, a method called FairAdj [32] has revealed that unfair connections within the graph structure may cause bias in predictions, and learns a fair graph structure to promote group fairness. Moreover, exploring the relations between the explainability and fairness of GNNs is also helpful for providing insights that may help to alleviate discrimination in the GNN context.

VII. ACCOUNTABILITY OF GNNS

Accountability refers to the extent to which people can trust GNNs by assessing a complex GNN system. In practice, GNN systems are used as black boxes that provide few transparency to either developers or users. For example, cloud-based GNN services perform their functions in the cloud [168]; their underlying operations cannot be directly accessed by end users. In the absence of proper accountability, users will not fully trust GNN service providers, since they have concerns due to frequent security incidents on the cloud [242]. In addition, there are large numbers of blocks in today's graph-based GNN systems and various roles in play when developing and maintaining such complex systems. It is necessary to identify who or what is responsible for each component when an error occurs. All the above concerns give rise to the requirement for accountability.

Due to the diverse range of application scenarios and research focuses, different researchers have proposed different definitions of accountability. In general, the concept of accountability encompasses a number of aspects [243], including: *Detection*: A system is accountable if it can provide a means to detect violations or misbehaviour; *Evidence*: Accountability refers to the ability to present evidence that might convince a third party. *Identification and association*: Accountability is the property through which the state and actions of identities can be associated in a way that enables any misbehaviour to be detectable, provable and undeniable; *Answerability*: Accountability is the notion of being liable to answer for conduct (e.g., incident management and remediation); *Blame*: Accountability refers to the ability to assign responsibility for actions based on additional analysis beyond the direct mapping of identification; *Punishment*: A system is accountable if it can impose sanctions in the event that responsibilities are not met.

In light of the above definitions and the guidelines provided by the European Commission [244], we outline three common objectives that should be referenced when building an accountable GNN: (1) design a set of conformable assessment and certification processes throughout the entire development cycle of GNN systems, that enable violations or misbehaviour to be detected and associated with specific identities (*Detection, Identification and association, Blame*); (2) provide assurance of auditability, e.g., by providing open access to event logs in order to provide a record that will be convincing to third parties (*Evidence*); (3) set up explainable coordinations such that humans will be capable of adjusting, remediating, and punishing the actions of a GNN with ease (*Answerability and Punishment*).

From a technical perspective, the majority of existing studies concerning GNN accountability focus primarily on achieving the first objective, namely proposing conformable assessments of GNNs to facilitate the detection of violations. In this section, we will present some of these methods, then go on to propose potential directions for research into enhancing accountability.

A. Methods

According to the type of violation involved, existing technologies for improving GNN accountability can be divided into two categories: those that detect violations of utility, and those that detect violations of security. A violation of utility indicates that the GNN is not performing as expected (particularly with regard to its accuracy). A violation of this kind can be detected and analysed via benchmarking frameworks, which can be used to assess the GNN performance (i.e., to determine whether it is performing as expected) and to identify the corresponding impact factors (associating the violation with an identity). Moreover, a violation of security refers to a breach of security policies [75] (confidentiality, integrity, and availability) and can be detected by security evaluation techniques. Note that there may be other methods for detecting these two types of violations, as well as other violations in the GNN context (e.g., violations of the law or ethics).

Benchmarking. To comprehensively assess GNNs over their entire life cycle, it is necessary to propose standardised evaluation methods for every step of GNN development. Benchmarking frameworks can be created to determine whether GNN models have been properly constructed, trained and validated. Categorised by the different stages of GNN development, they are listed as follows:

- *Architecture design.* Due to the wide variety of different applications and graph types, there are numerous kinds of GNN models and architectures. Thus, several research projects attempt to develop benchmarking frameworks to assist developers in selecting the appropriate model hyperparameters [245], [34].
- *Model training.* Due to the lack of transparency in model training, a set of works [246], [247] have attempted to examine how various training strategies (e.g., training hyperparameters, optimisation problem constructions) can impact the performance of GNNs.
- *Model validation.* Following model training, it is necessary to evaluate the final performance of GNNs. A set of studies [34], [248] have sought to develop comprehensive frameworks for evaluating GNN models, which can assist users and developers in validating their GNNs.

Security Evaluation. To evaluate the security properties of a GNN (i.e., confidentiality, integrity, availability), several security evaluation techniques can be applied. While these techniques have previously been employed in general-purpose computing systems, most existing studies on ML-based systems have focused on integrity verification. Due to the similarities between the development and implementation processes of GNNs and DNNs, these methods can be directly applied or easily adapted to GNNs despite having been developed for DNNs. Depending upon the objective of the verification, there are two types of integrity verification: data integrity verification and procedure integrity verification.

- *Data integrity.* Data (including graphs as well as models) is one of the key components of GNN systems. Most current studies focus on verifying the integrity of models during inference. Javaheripi *et al.* [249] propose to verify the integrity during running time by means of a unique hash signature extracted from the benign model prior to its deployment. For their part, He *et al.* [250] construct sensitive-sample fingerprints whose outputs can clearly reflect even small changes to the model.
- *Procedure integrity.* Aside from data in GNNs, it is also imperative to ensure the integrity of each step in the system development process. Most current works focus on verifying the training procedure [251], [252]. For example, Jia *et al.* [251] combine studies in *proof-of-work* and *verified computations* to verify the training process by analysing the initial models, intermediate models, and final trained models.

B. Future Directions of GNN Accountability

Despite the urgent demands to address accountability in GNN systems, there have been few attempts to do so. On this subject, several future research directions should be considered: (1) Detection and association of other types of violations. Current research into building accountable GNNs focuses only on detecting violations of utility and security. However, there are other factors that should be taken into consideration, such as ethical, legal, and environmental violations. It is also necessary to conduct consistent assessments of these violations and evaluate their effects. (2) Verification of other components of GNN systems. The verification techniques introduced above tend to focus on either the training process or trained GNN models. Nevertheless, the security of other components (e.g., the integrity of graph data, the inference process of GNN models, etc.) should also be guaranteed. (3) Development of mechanisms to achieve other objectives (e.g., providing auditability and explainable coordination) of building accountable GNNs.

VIII. Environmental Well-being of GNNs

Competent GNNs also should conform to the fundamental values of the society in which they are deployed. A shared

expectation of societies with diverse cultural backgrounds is that GNNs should perform competently and efficiently across their range of applications. One study on energy consumption [253] shows that training a common NLP model produces the same amount of carbon dioxide as a human would produce in seven years. To reduce the environmental impacts of GNNs and ensure their trustworthiness, it is crucial to explore the environmental well-being of GNNs.

In this survey, our account of the environmental well-being of GNNs focuses on methods that can improve GNN efficiency when they are faced with various practical challenges. Some representative efficiency-related challenges are as follows: (1) Given the explosive growth of graph data, large-scale datasets [254] pose a significant challenge to efficient GNN execution in terms of both training and inference. (2) Deeper or more complex GNN architectures [18] have been proposed to obtain better task performance, overcome over-smoothing, or reduce over-fitting [255]. However, the sizes of these models challenge their deployment on edge devices (e.g., autonomous vehicles [71]) with limited computational resources (e.g., memory) [256]. (3) Due to the unique characteristics of graph data (e.g., irregularity), the study and deployment of GNNs would benefit significantly from specially designed and efficient software frameworks and hardware accelerators [69].

In this section, we review recent methods for improving the efficiency of GNNs, which is regarded as aligning GNN research with social values regarding environmental well-being. Generally, the efficiency improvement is evaluated with reference to time-related metrics (e.g., response latency or speedup rating [257], [258], throughput rating [259], [260], communication time [261]), energy-related metrics (e.g., nodes-per-Joule [36], energy consumption [262]), or resource-related metrics (e.g., memory footprint [72], cache access performance, and peak memory usage [263]). Existing methods include scalable GNN architectures and efficient data communication, model compression methods, efficient frameworks and accelerators.

A. Methods

1) Scalable GNN Architectures and Efficient Data Communication: Scalable GNN architectures are GNNs that can efficiently process graphs on a giant scale. Recently, giant-scale graphs obtained from real-world scenarios have made the application of GNNs more challenging. For example, even an on-line gaming graph called Friendster might contain 65.6 million nodes and over 1.8 billion edges [264]. To handle large-scale graphs, some studies have explored sampling methods [48], [265], scalable architectures [49], [18], and industrial applications with GNNs [5], [266]. Moreover, some GNNs suffer from inefficient data loading. For example, data loading occupies 74% of the whole training time for GCN [260]. A method called PaGraph [260] analyses pipeline bottlenecks of GNNs and proposes a GPU cache policy to reduce the time consumption associated with moving data from CPU to GPU.

2) Model Compression Methods: Due to the irregularity of graph data, the exponential expansion of computational graphs challenges the efficiency of GNNs when stacking GNN layers, which limits the applicability of deep GNNs in real-time systems. To balance the efficiency and competence of GNNs, recent studies have introduced knowledge distillation, model pruning, reducing parameters, and model quantisation to GNNs for compressing GNN models.

Knowledge distillation aims to utilise competent deep GNNs to train lightweight GNNs that can achieve similar performance [267], [268]. For example, a method called TinyGNN [257] is proposed to learn local structure knowledge from a GNN model by distilling neighbour nodes. Another effective approach for acceleration is model pruning. One representative method, called UGS [269], introduces the lottery ticket hypothesis to GNNs, which can prune and accelerate GNNs on large-scale graphs. Moreover, reducing the size of GNN parameters can also improve their efficiency. For example, a method called GNF [72] employs normalising flows to achieve reversible GNNs, which reduces the memory footprint of GNNs and improves their practicability. Additionally, the inference of GNNs can be accelerated with the help of low-precision integer arithmetic (i.e., model quantisation) [270]. Recently, an architecture-agnostic method called Degree-Quant [258] has demonstrated how to implement quantization-aware training, which can achieve efficient and competent GNNs.

3) Efficient Frameworks and Accelerators: The computing of GNNs presents a series of efficiency challenges, including computational dependence on input graphs, intertwined computation between dense and sparse operations, and a diversity of applications and tasks [69]. To this end, several software (SW) frameworks and hardware (HW) accelerators have been proposed to facilitate the development of efficient GNNs [69].

Different graph SW frameworks have been developed to provide efficient interfaces for GNN implementations. These frameworks are designed for sparse operations or efficient execution on various graphs (e.g., spatio-temporal graphs) in a multi-GPU environment. Typical SW frameworks [69] include PyTorch Geometric (PyG) [271], Deep Graph Library (DGL) [272], etc. In addition to the above SW frameworks, HW accelerators have emerged to streamline the execution of GNNs. These HW accelerators are energy-efficient and increase the operating speed of GNNs by two to three times [69]. Typical HW accelerators include the EnGN [273], HyGCN [262], etc. In addition, some studies have focused on analysing the efficiency bottleneck of GNNs [274] and an SW-HW co-design for the accelerator [275].

B. Future Directions of Environmental Well-being in GNNs

Exploration of Efficient GNNs. Although improving the efficiency of AI systems has attracted the attention of AI practitioners, the exploration of efficient GNNs has been limited. For various types of graph data (e.g., spatio-temporal graph) and applications, one direction is to analyse the efficiency bottleneck of current GNNs and then improve them or explore new and more efficient operations in GNNs. In GNN deployment, accelerating the inference of GNNs [259], [263] is a practical approach for achieving more efficient GNNs. Compressing models and accelerating the inference of GNNs

TABLE VII
REPRESENTATIVE METHODS FOR TRUSTWORTHY GNNS.

	Robustness	Explainability	Privacy	Fairness	Account-ability	Environmental Well-being	Others
Robustness	[44], [52], [112], [87], [108], [106], [105], [89], [118], [276]★						
Explainability	[95]▲, [128]▲	[145], [58], [22], [56], [132], [150], [160], [165], [161], [151]					
Privacy			[174], [60], [208], [277], [202], [211], [45], [278]				
Fairness	[64]▲		[235]•, [21]•	[235], [33], [32], [64], [65], [66], [21]			
Accountability					[34]		
Environmental Well-being	[279]•, [106]•, [280]•, [108]•, [109]•	[281]▲	[61]•		[282]•	[49], [283], [284], [285], [286], [262], [273], [48], [258], [18], [265]	
Others						[279]•	[74]

★ The references at diagonal positions indicate representative studies for one aspect of trustworthy GNNs. The references at non-diagonal positions represent recent studies related to complex cross-aspect relations, where methods/advancements from one aspect can affect another aspect of trustworthy GNNs.

▲, • In this table, ▲ indicates the studies showing *how the methods from one aspect of trustworthy GNNs are adapted to address objectives in other aspects*, and • marks the studies that discuss *why advancing one aspect of trustworthy GNNs can promote or inhibit other aspects*. Moreover, references **with/without underline** indicate studies that present **potential/clear** relations; see Section IX for more details.

will directly contribute to the environmental well-being of GNNs. Moreover, introducing efficiency considerations into GNN training [258] will also promote the development of efficient GNNs. Adaptive architecture designs aimed at saving energy [287], [43] have been proposed in the context of CNN models; thus, it would also be worthwhile to analyse the energy consumption of different operations in GNNs and their operational complexity, since these factors may contribute to efficient GNN architectures by Graph NAS [288].

Accelerators for GNNs. Current research into accelerators for GNNs focuses on designing software frameworks and hardware separately. We argue that the acceleration of GNNs should take the whole pipeline of GNNs—consisting of the modelling, applications, complexity, algorithms, accelerators, HW/SW requirements, and data flow—into account. By identifying the actual requirements of GNNs in applications (e.g., quick response in real-time systems, limited memory footprint in edge computing terminals) and the bottlenecks in the execution computation of GNNs (e.g., data flow between CPU and GPU), the SW-HW co-design [289] is a practical and promising direction for future research aimed at achieving the environmental well-being of GNNs.

IX. RELATIONS BETWEEN ASPECTS OF TRUSTWORTHY GNNS

Most studies in Sections III-VIII focus on one specific aspect of trustworthy GNNs. However, when building trustworthy GNN systems, it is necessary to take complicated cross-aspect relationships into account; this is because in practical development and deployment, the above six aspects of GNNs are highly related, and an interplay (e.g., accordance and trade-off) exists among them. To thoroughly understand and build trustworthy GNN systems, it is both inevitable and imperative to explore and study the complex relations between these aspects.

In this section, we present existing studies of these relations (as shown in Table VII) by with focuses on the following: (1) *how the methods from one aspect of trustworthy GNNs are adapted to address objectives in other aspects*, and (2) *why advancing one aspect of trustworthy GNNs can promote or inhibit other aspects*.

Explainability and Robustness. The methods developed to explore GNN explainability can be used to implement adversarial attacks and defences on GNNs. Explanation methods reveal the behaviours of GNNs by identifying why a GNN

makes a specific prediction with regard to the current samples. Thus, they provide insights that support the design of attack and defence algorithms. For example, in explainability-based backdoor attacks [95], GNNExplainer is employed [22] to identify the importance of nodes and guide the selection of the injected backdoor trigger position. Moreover, defenders can also use explainers for GNNs to identify the potential locations of malicious edge perturbations [128].

Robustness, Fairness, and Privacy. Methods for improving the robustness of GNNs can be adapted to aid in improving GNN fairness. For example, adversarial training is a typical method for achieving robust GNNs by perturbing training data. Adopting a related perspective, a method called NIFTY [64] follows a strategy similar to adversarial training to perturb sensitive attributes in order to counteract the unfair GNN predictions regarding these attributes. The privacy of GNNs also benefits from the improved GNN fairness. For example, fair representation learning methods designed to improve the fairness of GNNs also reduce the risk of private information leakage from sensitive attributes [21], [235], as it is difficult to infer these attributes from the learned representations.

Environmental Well-being and Others. Compared with efforts to improve other aspects, methods targeting environmental well-being exhibit more diversity, which results in the existence of multiple relations between environmental well-being and other aspects. For example, methods designed to promote the environmental well-being (i.e., model compression methods) of GNNs can also facilitate exploring the explainability of GNNs. A knowledge distillation method called CPF [281] can be used to help build surrogate models that facilitate GNN explanation. Moreover, environmental well-being has complex advancement-level relations with other aspects of trustworthy GNNs. Several examples are presented below.

Achieving environmental well-being (i.e., model compression) of GNNs may contribute to the robustness. The quantisation of GNNs, a general approach for improving efficiency, also makes them more robust. Generally, quantised deep learning models are more robust to noise and adversarial attacks [290]. The quantisation of models can be considered as adding extra terms in objective functions, which reduces over-fitting risks and increases the generalisation ability and robustness of models. Recent research shows that such insights can be also applied to GNNs [279].

Better accountability techniques also improve the environmental well-being of GNNs. Specifically, accountability in the GNN context involves implementing a series of processes to assess the GNN systems. Meanwhile, most of these studies also aim to assess GNN systems efficiently. The improvement of accountability techniques can thus also promote a higher level of GNN environmental well-being. For example, Lacombe *et al.* [282] develop a framework that provides developers with valuable knowledge regarding GNN training. Specifically, it indicates whether the training of GNNs is processing as expected, which is helpful for stopping futile training and reducing energy consumption.

Building robust GNNs can also impact the environmental well-being of GNNs. Graph structure learning is a typical robustness enhancement method for GNNs, but its clean graph learning process (e.g., training an extra graph generation module [106], [280], [291]) increases the development cost of GNNs. Moreover, introducing specially designed operations in GNN architectures [108], [109] improves GNN robustness; however, more operation efforts are needed to develop and implement these modified GNNs. These robustness-oriented efforts lead to higher energy costs, which are detrimental to the environmental well-being of trustworthy GNNs.

The implementation of privacy techniques for GNNs also increases the costs of developing and implementing GNN systems, which hinders the environmental well-being of trustworthy GNNs. Generally speaking, achieving privacy in the GNN context necessitates the addition of extra processes or specific designs, which in turn increases the amount of development effort required. For example, a privacy-preserving GNN architecture reduces the leakage of sensitive attributes but increases GNN training costs [61].

X. Conclusion and Trending Research Directions

Trustworthy GNNs is a thriving research field with a wide range of methodologies and applications. This survey summarises current advancements and trends, which is expected to facilitate and advance future research into, and implementation of, trustworthy GNNs.

From the perspective of technology, this survey presents a roadmap towards comprehensively building trustworthy GNNs. We first define trustworthy GNNs and depict their characteristics by categorising principles in the existing literature related to trustworthiness. Next, we review recent advancements in trustworthy GNNs from the aspects of robustness, explainability, privacy, fairness, accountability, and environmental well-being. For each aspect, we introduce the basic concepts, comprehensively summarise the existing methods for building trustworthy GNNs, and then point out potential future research directions related to these aspects. Additionally, we emphasise that cross-aspect study is vital for achieving trustworthy GNNs. We present complex relations between aspects by introducing methodology transfer and discussing the points of agreement and conflict between each aspect.

While trustworthy GNNs have attracted increasing attention, applying them to real-world applications still poses many challenges; these range from the design philosophy of GNNs to the conceptual understanding of trustworthiness, and from the diversified relations between aspects of trustworthy GNNs to the practicability of the proposed methods in specific real-world applications. Moving away from the future directions suggested for specified aspects of trustworthy GNNs in Sections III-VIII, we here highlight five trending directions with the potential to fill the research gap and boost the industrialisation of trustworthy GNNs by considering the concept as a whole.

Shift to Trustworthy GNNs. Building trustworthy GNNs requires researchers to embrace the trustworthiness-related concepts and characteristics introduced in this survey and shift their focuses from performance-oriented GNNs to trustworthy GNNs. With regard to achieving this shift, introducing trustworthiness into the design philosophy of GNNs is a challenging yet promising direction. An example of incorporating

explainability is a method called GIB [144], which can simultaneously recognise an information bottleneck subgraph (as an explanation) for the input graph and cooperate with other GNN backbones in graph classification. As an example of incorporating fairness, RawlsGCN [241] alleviates the degree-related unfairness by introducing the Rawlsian difference principle [292] (i.e., maximising the welfare of the worst-off groups [241]) to GCN. Moreover, some open problems need to be solved if trustworthy GNN systems are to be realised; for example, that of how to balance the trade-off between certain aspects (e.g., robustness and environmental well-being) in specified applications (e.g., autonomous vehicles [71]). Thus, shifting to trustworthy GNNs and solving these problems are fundamental for building trustworthy GNN systems.

Other Aspects of Trustworthy GNNs. The content of trustworthy GNNs goes beyond the six aspects discussed in this survey. For example, generalisation ability is considered to be a necessary aspect of trustworthy systems [11]. Current research into the extrapolation of GNNs [74] aims to reveal the relationship between tasks and aggregation functions, which will promote further study of the generalisation of GNNs and guide the design of trustworthy GNNs. Therefore, studying the proper handling and management of the trustworthiness-related principles outlined in Table I (e.g., safety and controllability, respect for autonomy) is another vital research direction in building human-centred and trustworthy GNN systems. The open framework proposed in this survey (Fig. 1) will hopefully be able to incorporate aspects of trustworthiness that emerge in the future, thereby expanding the domain of trustworthy GNNs and enhancing the practical implementation of trustworthy GNN systems.

Diversified Relations. This survey touches on only a small fraction of the intricate cross-aspect relations. It will therefore be crucial to investigate other cross-aspect relations like explainability and privacy in future work. Moreover, these cross-aspect relations are not only intricate, but also exist at multiple levels. For example, counterfactual fairness is conceptually similar to robustness, as both of them expect GNNs to make unchanged predictions when some attributes of the same sample are changed. This clearly suggests that methods for improving GNN robustness can also be used to improve the fairness of GNNs to a certain extent (e.g., by introducing adversarial training on GNNs with respect to sensitive attributes [64]). Hence, exploring the diversified relations at different levels (e.g., concepts, methods, and advancements) is also essential to comprehensively understanding and practically building trustworthy GNNs.

Model-agnostic Methods. Many methods for trustworthy GNNs require custom-designed network architectures, which makes them unable to improve the trustworthiness of the target GNNs without modifying their architectures. This weakness severely limits the practicability of these trust-enhancing methods for GNNs. By contrast, model-agnostic methods can increase the trustworthiness of GNNs simply by being plugged in to any phase of the system pipeline (i.e., pre-processing, in-processing, and post-processing). Another advantage of these methods is that they can be combined to serve as building blocks for trustworthy GNNs. Thus, the design of model-agnostic methods is a promising direction for trustworthy GNNs.

Technology Ecosystem for Trustworthy GNNs. As an emerging field, research into trustworthy GNNs is inseparable from the surrounding technological ecosystem, including the tools, datasets, metrics, pipelines, etc. Current tools (e.g., AI 360 tools from IBM [293]) are mainly used to improve the credibility of general machine learning models. However, it may not be easy to apply these tools directly to GNNs owing to the unique characteristics of graphs. For example, updating the embedding of nodes in GNNs usually relies on the edges between nodes. The presence of these edges causes the relationship between nodes to break the independent and identically distributed assumption in general machine learning models, which constitutes a challenge for studying the individual fairness of GNNs. Moreover, various datasets [254], [294], metrics [295], standardised evaluation pipelines [245], [34], and software platforms [296], [297] that are suitable for miscellaneous applications will also be required if efforts to develop trustworthy GNNs are to be successfully evaluated. Therefore, developing a technological ecosystem that supports trustworthiness is an essential step in researching and industrialising trustworthy GNNs.

ACKNOWLEDGMENTS

This research was supported in part by the Australian Research Council (ARC) under a Future Fellowship No. FT210100097.

APPENDIX A
DATASETS FOR TRUSTWORTHY GNNS

Although some aspects of trustworthy GNNs (e.g., robustness) have been continually explored in recent years, the study of other aspects remains in the initial stages. Current studies on robustness, privacy, accountability and environmental well-being can be implemented by evaluations on common graph datasets (e.g., TUDataset [294], OGB [298]); however, when studying the explainability and fairness of GNNs, general graph datasets are not suitable because they do not contain ground-truth explanations and sensitive attributes, respectively. In this section, we briefly review datasets for evaluating the explainability and fairness of GNNs.

Explainability. The biggest challenge associated with evaluating explanation methods for GNNs is the lack of ground truth datasets. Due to the complexity of graph data in real-world applications, it is difficult for humans to understand which components of graph data are the ground-truth reasons for the predictions made by GNNs. Accordingly, some methods use synthetic data to evaluate the quality of explanations. In these datasets, different graph motifs are attached to the base graph. To facilitate easy evaluation of these explanations, the node labels and graph labels are defined by their role in the synthetic dataset and the motif type they contain, respectively. The most widely used datasets of this kind are BA-shapes, BA-Community, Tree-Cycle, Tree-Grids, BA-2Motifs [29], and Syn-Cora [141].

Although it is human-intelligible to employ synthetic datasets in evaluation, these synthetic datasets cannot provide adequate benchmarks for complex and various real-world applications. Accordingly, several common real-world datasets are also used to evaluate explanations. In biochemistry research, the functionality (e.g., mutagenic effect) of some molecules is determined by their components. Datasets used for related research include MUTAG, BBBP and Tox21 [294], [29]. In natural language processing, three sentiment graph datasets are created as sentiment graph data from text sentiment analysis data. They are Graph-SST2, Graph-SST5, and Graph-Twitter [29]. In natural language processing, three sentiment graph datasets are created from text sentiment analysis data, namely Graph-SST2, Graph-SST5, and Graph-Twitter [29].

Fairness. Current methods use various datasets to evaluate the fairness of GNNs. For similarity-based fairness (e.g., individual fairness), datasets for node classification and link prediction are used to conduct fairness evaluation [66]. For fairness concerning protected or sensitive attributes (e.g., group fairness), these methods choose datasets that contain sensitive or protected information, such as that pertaining to nationality or region [21]. Some typical datasets are Freebase 15k-237, MovieLens-1M, Reddit [235], Pokec, NBA [21], German Credit, Recidivism, Credit Defaulter [64], Oklahoma97, UNC28 [33], Dutch School, Facebook, and Google+ [299].

REFERENCES

[1] M. Osman, "Wild and interesting facebook statistics and facts (2021)," Jan 2021, accessed December 21, 2021. [Online]. Available: https://kinsta.com/blog/facebook-statistics/

[2] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and S. Y. Philip, "A comprehensive survey on graph neural networks," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, no. 1, pp. 4–24, 2021.

[3] L. Ruiz, F. Gama, and A. Ribeiro, "Graph neural networks: Architectures, stability, and transferability," *Proceedings of the IEEE*, vol. 109, no. 5, pp. 660–682, 2021.

[4] R. Zhu, K. Zhao, H. Yang, W. Lin, C. Zhou, B. Ai, Y. Li, and J. Zhou, "Aligraph: A comprehensive graph neural network platform," *Proceedings of the VLDB Endowment*, vol. 12, no. 12, pp. 2094–2105, 2019.

[5] A. Pal, C. Eksombatchai, Y. Zhou, B. Zhao, C. Rosenberg, and J. Leskovec, "Pinnersage: Multi-modal user embedding framework for recommendations at pinterest," in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 2020, pp. 2311–2320.

[6] A. Sanchez-Gonzalez, J. Godwin, T. Pfaff, R. Ying, J. Leskovec, and P. W. Battaglia, "Learning to simulate complex physics with graph networks," in *Proceedings of the International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, vol. 119. PMLR, 2020, pp. 8459–8468.

[7] R. Abbasi, M. Ackermann, J. Adams, J. Aguilar, M. Ahlers, M. Ahrens, C. M. Alispach, A. A. Alves Junior, N. M. Amin, R. An *et al.*, "Study of mass composition of cosmic rays with icetop and icecube," in *Proceedings of the International Cosmic Ray Conference*, 2021.

[8] H. Yuan, J. Zheng, Q. Ye, Y. Qian, and Y. Zhang, "Improving fake news detection with domain-adversarial and graph-attention neural network," *Decision Support Systems*, vol. 151, p. 113633, 2021.

[9] W. Jin, J. M. Stokes, R. T. Eastman, Z. Itkin, A. V. Zakharov, J. J. Collins, T. S. Jaakkola, and R. Barzilay, "Deep learning identifies synergistic drug combinations for treating COVID-19," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 118, no. 39, 2021.

[10] H. Liu, Y. Wang, W. Fan, X. Liu, Y. Li, S. Jain, A. K. Jain, and J. Tang, "Trustworthy AI: A computational perspective," *CoRR*, vol. abs/2107.06641, 2021.

[11] B. Li, P. Qi, B. Liu, S. Di, J. Liu, J. Pei, J. Yi, and B. Zhou, "Trustworthy AI: from principles to practices," *CoRR*, vol. abs/2110.01167, 2021.

[12] K. R. Varshney, "Trustworthy machine learning," http://www.trustworthymachinelearning.com, 2021.

[13] M. Brundage, S. Avin, and J. W. et al., "Toward trustworthy AI development: Mechanisms for supporting verifiable claims," *CoRR*, vol. abs/2004.07213, 2020.

[14] "Governance principles for the new generation artificial intelligence–developing responsible artificial intelligence," https://www.chinadaily.com.cn/a/201906/17/WS5d07486ba3103dbf14328ab7.html, 2019, accessed December 19, 2021.

[15] "The montreal declaration of responsible ai," https://www.montrealdeclaration-responsibleai.com/the-declaration, 2017, accessed December 19, 2021.

[16] K. Xu, W. Hu, J. Leskovec, and S. Jegelka, "How powerful are graph neural networks?" in *International Conference on Learning Representations*, 2019.

[17] D. Chen, Y. Lin, W. Li, P. Li, J. Zhou, and X. Sun, "Measuring and relieving the over-smoothing problem for graph neural networks from the topological view," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2020.

[18] G. Li, M. Müller, B. Ghanem, and V. Koltun, "Training graph neural networks with 1000 layers," in *Proceedings of the International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, vol. 139. PMLR, 2021, pp. 6437–6449.

[19] Y. Liu, M. Jin, Y. Zheng, S. Pan, C. Zhou, F. Xia, and P. S. Yu, "Graph self-supervised learning: A survey," *IEEE Transactions on Knowledge and Data Engineering*, 2022.

[20] Y. Zhu, Y. Lai, K. Zhao, X. Luo, M. Yuan, J. Ren, and K. Zhou, "Binarizedattack: Structural poisoning attacks to graph-based anomaly detection," in *The annual IEEE International Conference on Data Engineering*. IEEE, 2022.

[21] E. Dai and S. Wang, "Say no to the discrimination: Learning fair graph neural networks with limited sensitive attribute information," in *Proceedings of the ACM International Conference on Web Search and Data Mining*. ACM, 2021, pp. 680–688.

[22] Z. Ying, D. Bourgeois, J. You, M. Zitnik, and J. Leskovec, "Gnnexplainer: Generating explanations for graph neural networks," in *Advances in Neural Information Processing Systems*, 2019, pp. 9240–9251.

[23] O. L. Dictionaries, "Definition of trustworthy," accessed December 19, 2021. [Online]. Available: https://www.oxfordlearnersdictionaries.com/definition/american_english/trustworthy

[24] J. D. Lewis and A. Weigert, "Trust as a social reality," *Social forces*, vol. 63, no. 4, pp. 967–985, 1985.

[25] C. Juvekar, V. Vaikuntanathan, and A. P. Chandrakasan, "GAZELLE: A low latency framework for secure neural network inference," in *USENIX Security Symposium*. USENIX Association, 2018, pp. 1651–1669.

[26] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proceedings of the International Conference on Artificial Intelligence and Statistics*, ser. Proceedings of Machine Learning Research, vol. 54. PMLR, 2017, pp. 1273–1282.

[27] M. Abadi, A. Chu, I. J. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang, "Deep learning with differential privacy," in *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2016, pp. 308–318.

[28] W. Jin, Y. Li, H. Xu, Y. Wang, S. Ji, C. Aggarwal, and J. Tang, "Adversarial attacks and defenses on graphs," *ACM SIGKDD Explorations Newsletter*, vol. 22, no. 2, pp. 19–34, 2020.

[29] H. Yuan, H. Yu, S. Gui, and S. Ji, "Explainability in graph neural networks: A taxonomic survey," *CoRR*, vol. abs/2012.15445, 2020.

[30] C. Agarwal, M. Zitnik, and H. Lakkaraju, "Towards a rigorous theoretical analysis and evaluation of GNN explanations," *CoRR*, vol. abs/2106.09078, 2021.

[31] C. He, K. Balasubramanian, E. Ceyani, Y. Rong, P. Zhao, J. Huang, M. Annavaram, and S. Avestimehr, "Fedgraphnn: A federated learning system and benchmark for graph neural networks," *CoRR*, vol. abs/2104.07145, 2021.

[32] P. Li, Y. Wang, H. Zhao, P. Hong, and H. Liu, "On dyadic fairness: Exploring and mitigating bias in graph connections," in *International Conference on Learning Representations*. OpenReview.net, 2021.

[33] J. Kang, J. He, R. Maciejewski, and H. Tong, "Inform: Individual fairness on graph mining," in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 2020, pp. 379–389.

[34] W. Zhao, D. Zhou, X. Qiu, and W. Jiang, "A pipeline for fair comparison of graph neural networks in node classification tasks," *CoRR*, vol. abs/2012.10619, 2020.

[35] A. Auten, M. Tomei, and R. Kumar, "Hardware acceleration of graph neural networks," in *ACM/IEEE Design Automation Conference*. IEEE, 2020, pp. 1–6.

[36] Z. Zhou, B. Shi, Z. Zhang, Y. Guan, G. Sun, and G. Luo, "Blockgnn: Towards efficient GNN acceleration using block-circulant weight matrices," in *ACM/IEEE Design Automation Conference*. IEEE, 2021, pp. 1009–1014.

[37] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," in *International Conference on Learning Representations*, 2015.

[38] M. T. Ribeiro, S. Singh, and C. Guestrin, ""why should I trust you?": Explaining the predictions of any classifier," in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 2016, pp. 1135–1144.

[39] K. Simonyan, A. Vedaldi, and A. Zisserman, "Deep inside convolutional networks: Visualising image classification models and saliency maps," in *International Conference on Learning Representations*, 2014.

[40] R. Shokri, M. Stronati, C. Song, and V. Shmatikov, "Membership inference attacks against machine learning models," in *IEEE Symposium on Security and Privacy*. IEEE Computer Society, 2017, pp. 3–18.

[41] N. Mehrabi, F. Morstatter, N. Saxena, K. Lerman, and A. Galstyan, "A survey on bias and fairness in machine learning," *ACM Computing Surveys*, vol. 54, no. 6, pp. 115:1–115:35, 2021.

[42] M. Wieringa, "What to account for when accounting for algorithms: a systematic literature review on algorithmic accountability," in *Proceedings of the Conference on Fairness, Accountability, and Transparency*. ACM, 2020, pp. 1–18.

[43] T. Yang, Y. Chen, and V. Sze, "Designing energy-efficient convolutional neural networks using energy-aware pruning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. IEEE Computer Society, 2017, pp. 6071–6079.

[44] D. Zügner, A. Akbarnejad, and S. Günnemann, "Adversarial attacks on neural networks for graph data," in *Proceedings of the International Joint Conference on Artificial Intelligence*. ijcai.org, 2019, pp. 6246–6250.

[45] X. He, J. Jia, M. Backes, N. Z. Gong, and Y. Zhang, "Stealing links from graph neural networks," in *USENIX Security Symposium*. USENIX Association, 2021, pp. 2669–2686.

[46] X. He, R. Wen, Y. Wu, M. Backes, Y. Shen, and Y. Zhang, "Node-level membership inference attacks against graph neural networks," *CoRR*, vol. abs/2102.05429, 2021.

[47] B. Wu, X. Yang, S. Pan, and X. Yuan, "Adapting membership inference attacks to GNN for graph classification: Approaches and implications," in *IEEE International Conference on Data Mining*. IEEE Computer Society, 2021.

[48] H. Zeng, H. Zhou, A. Srivastava, R. Kannan, and V. K. Prasanna, "Graphsaint: Graph sampling based inductive learning method," in *International Conference on Learning Representations*. OpenReview.net, 2020.

[49] F. Wu, A. H. S. Jr., T. Zhang, C. Fifty, T. Yu, and K. Q. Weinberger, "Simplifying graph convolutional networks," in *Proceedings of the International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, vol. 97. PMLR, 2019, pp. 6861–6871.

[50] N. A. Asif, Y. Sarker, R. K. Chakrabortty, M. J. Ryan, M. H. Ahamed, D. K. Saha, F. R. Badal, S. K. Das, M. F. Ali, S. I. Moyeen, M. R. Islam, and Z. Tasneem, "Graph neural network: A comprehensive review on non-euclidean space," *IEEE Access*, vol. 9, pp. 60 588–60 606, 2021.

[51] Y. Ma and J. Tang, *Deep learning on graphs*. Cambridge University Press, 2021.

[52] H. Dai, H. Li, T. Tian, X. Huang, L. Wang, J. Zhu, and L. Song, "Adversarial attack on graph structured data," in *Proceedings of the International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, vol. 80. PMLR, 2018, pp. 1123–1132.

[53] A. Bojchevski and S. Günnemann, "Adversarial attacks on node embeddings via graph poisoning," in *Proceedings of the International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, vol. 97. PMLR, 2019, pp. 695–704.

[54] K. Zhou, T. P. Michalak, M. Waniek, T. Rahwan, and Y. Vorobeychik, "Attacking similarity-based link prediction in social networks," in *Proceedings of the International Conference on Autonomous Agents and MultiAgent Systems*. International Foundation for Autonomous Agents and Multiagent Systems, 2019, pp. 305–313.

[55] P. Dey and S. Medya, "Manipulating node similarity measures in networks," in *Proceedings of the International Conference on Autonomous Agents and MultiAgent Systems*. International Foundation for Autonomous Agents and Multiagent Systems, 2020, pp. 321–329.

[56] D. Luo, W. Cheng, D. Xu, W. Yu, B. Zong, H. Chen, and X. Zhang, "Parameterized explainer for graph neural network," in *Advances in Neural Information Processing Systems*, 2020.

[57] B. Sánchez-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 *Advances in Neural Information Processing Systems*, 2020.

[58] P. E. Pope, S. Kolouri, M. Rostami, C. E. Martin, and H. Hoffmann, "Explainability methods for graph convolutional neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. Computer Vision Foundation / IEEE, 2019, pp. 10 772–10 781.

[59] G. Jaume, P. Pati, B. Bozorgtabar, A. Foncubierta, A. M. Anniciello, F. Feroce, T. Rau, J. Thiran, M. Gabrani, and O. Goksel, "Quantifying explainers of graph neural networks in computational pathology," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2021.

[60] B. Wu, X. Yang, S. Pan, and X. Yuan, "Model extraction attacks on graph neural networks: Taxonomy and realization," in *ACM Asia Conference on Computer and Communications Security*. ACM, 2022.

[61] B. Wang, J. Guo, A. Li, Y. Chen, and H. Li, "Privacy-preserving representation learning on graphs: A mutual information perspective," in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 2021, pp. 1667–1676.

[62] I. Hsieh and C. Li, "Netfense: Adversarial defenses against privacy attacks on neural networks for graph data," *CoRR*, vol. abs/2106.11865, 2021.

[63] E. Zheleva and L. Getoor, "Preserving the privacy of sensitive relationships in graph data," in *Proceedings of the ACM SIGKDD International Conference on Privacy, Security, and Trust in KDD*, ser. Lecture Notes in Computer Science, vol. 4890. Springer, 2007, pp. 153–171.

[64] C. Agarwal, H. Lakkaraju, and M. Zitnik, "Towards a unified framework for fair and stable graph representation learning," in *Uncertainty in Artificial Intelligence*. PMLR, 2021, pp. 2114–2124.

[65] I. Spinelli, S. Scardapane, A. Hussain, and A. Uncini, "Fairdrop: Biased edge dropout for enhancing fairness in graph representation learning," *IEEE Transactions on Artificial Intelligence*, 2021.

[66] Y. Dong, J. Kang, H. Tong, and J. Li, "Individual fairness for graph neural networks: A ranking based approach," in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 2021, pp. 300–310.

[67] F. Doshi-Velez, M. Kortz, R. Budish, C. Bavitz, S. Gershman, D. O'Brien, S. Schieber, J. Waldo, D. Weinberger, and A. Wood, "Accountability of AI under the law: The role of explanation," *CoRR*, vol. abs/1711.01134, 2017.

[68] Z. Zhang, P. Cui, and W. Zhu, "Deep learning on graphs: A survey," *IEEE Transactions on Knowledge and Data Engineering*, vol. 34, no. 1, pp. 249–270, 2022.

[69] S. Abadal, A. Jain, R. Guirado, J. López-Alonso, and E. Alarcón, "Computing graph neural networks: A survey from algorithms to accelerators," *ACM Computing Surveys*, vol. 54, no. 9, pp. 191:1–191:38, 2022.

[70] E. Dai, T. Zhao, H. Zhu, J. Xu, Z. Guo, H. Liu, J. Tang, and S. Wang, "A comprehensive survey on trustworthy graph neural networks: Privacy, robustness, fairness, and explainability," *CoRR*, vol. abs/2204.08570, 2022.

[71] J. Shao, H. Zhang, Y. Mao, and J. Zhang, "Branchy-gnn: A device-edge co-inference framework for efficient point cloud processing," in *IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2021, pp. 8488–8492.

[72] J. Liu, A. Kumar, J. Ba, J. Kiros, and K. Swersky, "Graph normalizing flows," in *Advances in Neural Information Processing Systems*, 2019, pp. 13 556–13 566.

[73] H. Li, X. Wang, Z. Zhang, and W. Zhu, "Out-of-distribution generalization on graphs: A survey," *CoRR*, vol. abs/2202.07987, 2022.

[74] K. Xu, M. Zhang, J. Li, S. S. Du, K. Kawarabayashi, and S. Jegelka, "How neural networks extrapolate: From feedforward to graph neural networks," in *International Conference on Learning Representations*, 2021.

[75] D. B. Parker, *Fighting computer crime - a new framework for protecting information*. Wiley, 1998.

[76] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl, "Neural message passing for quantum chemistry," in *Proceedings of the International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, vol. 70. PMLR, 2017, pp. 1263–1272.

[77] S. Pan, R. Hu, S. Fung, G. Long, J. Jiang, and C. Zhang, "Learning graph embedding with adversarial training methods," *IEEE Transactions on Cybernetics*, vol. 50, no. 6, pp. 2475–2487, 2020.

[78] S. Wan, Y. Zhan, L. Liu, B. Yu, S. Pan, and C. Gong, "Contrastive graph poisson networks: Semi-supervised learning with extremely limited labels," in *Advances in Neural Information Processing Systems*, 2021, pp. 6316–6327.

[79] D. Jin, Z. Yu, P. Jiao, S. Pan, P. S. Yu, and W. Zhang, "A survey of community detection approaches: From statistical modeling to deep learning," *IEEE Transactions on Knowledge and Data Engineering*, 2021.

[80] L. Wu, P. Cui, J. Pei, L. Zhao, and L. Song, "Graph neural networks," in *Graph Neural Networks: Foundations, Frontiers, and Applications*. Springer, 2022, pp. 27–37.

[81] X. Zou, Q. Zheng, Y. Dong, X. Guan, E. Kharlamov, J. Lu, and J. Tang, "TDGIA: effective injection attacks on graph neural networks," in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 2021, pp. 2461–2471.

[82] H. Zhang, B. Wu, X. Yang, C. Zhou, S. Wang, X. Yuan, and S. Pan, "Projective ranking: A transferable evasion attack method on graph neural networks," in *Proceedings of the ACM International Conference on Information & Knowledge Management*. ACM, 2021, pp. 3617–3621.

[83] Q. Zhou, L. Li, N. Cao, L. Ying, and H. Tong, "ADMIRING: adversarial multi-network mining," in *IEEE International Conference on Data Mining*. IEEE, 2019, pp. 1522–1527.

[84] K. Zhao, H. Zhou, Y. Zhu, X. Zhan, K. Zhou, J. Li, L. Yu, W. Yuan, and X. Luo, "Structural attack against graph based android malware detection," in *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2021, pp. 3218–3235.

[85] S. Freitas, D. Yang, S. Kumar, H. Tong, and D. H. Chau, "Graph vulnerability and robustness: A survey," *CoRR*, vol. abs/2105.00419, 2021.

[86] Y. Xia, J. Fan, and D. Hill, "Cascading failure in Watts–Strogatz small-world networks," *Physica A: Statistical Mechanics and its Applications*, vol. 389, no. 6, pp. 1281–1285, 2010.

[87] D. Zügner and S. Günnemann, "Adversarial attacks on graph neural networks via meta learning," in *International Conference on Learning Representations*. OpenReview.net, 2019.

[88] V. Gupta and T. Chakraborty, "VIKING: adversarial attack on network embeddings via supervised network poisoning," in *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, ser. Lecture Notes in Computer Science, vol. 12714. Springer, 2021, pp. 103–115.

[89] Y. Sun, S. Wang, X. Tang, T. Hsieh, and V. G. Honavar, "Adversarial attacks on graph neural networks via node injections: A hierarchical reinforcement learning approach," in *Proceedings of the ACM Web Conference*. ACM / IW3C2, 2020, pp. 673–683.

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

[91] S. Yu, J. Zheng, Y. Wang, J. Chen, Q. Xuan, and Q. Zhang, "Network embedding attack: An euclidean distance based method," in *MDATA: A New Knowledge Representation Model*, ser. Lecture Notes in Computer Science. Springer, 2021, vol. 12647, pp. 131–151.

[92] Z. Zhang, J. Jia, B. Wang, and N. Z. Gong, "Backdoor attacks to graph neural networks," in *Proceedings of the ACM Symposium on Access Control Models and Technologies*. ACM, 2021, pp. 15–26.

[93] J. Wang, M. Luo, F. Suya, J. Li, Z. Yang, and Q. Zheng, "Scalable attack on graph data by injecting vicious nodes," *Data Mining and Knowledge Discovery*, vol. 34, no. 5, pp. 1363–1389, 2020.

[94] Y. Ma, S. Wang, T. Derr, L. Wu, and J. Tang, "Graph adversarial attack via rewiring," in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 2021, pp. 1161–1169.

[95] J. Xu, M. Xue, and S. Picek, "Explainability-based backdoor attacks against graph neural networks," in *Proceedings of the ACM Workshop on Wireless Security and Machine Learning*. ACM, 2021, pp. 31–36.

[96] X. Zang, Y. Xie, J. Chen, and B. Yuan, "Graph universal adversarial attacks: A few bad actors ruin graph learning models," in *Proceedings of the International Joint Conference on Artificial Intelligence*. ijcai.org, 2021, pp. 3328–3334.

[97] J. Chen, Y. Chen, H. Zheng, S. Shen, S. Yu, D. Zhang, and Q. Xuan, "MGA: momentum gradient attack on network," *IEEE Transactions on Computational Social Systems*, vol. 8, no. 1, pp. 99–109, 2021.

[98] K. Xu, H. Chen, S. Liu, P. Chen, T. Weng, M. Hong, and X. Lin, "Topology attack and defense for graph neural networks: An optimization perspective," in *Proceedings of the International Joint Conference on Artificial Intelligence*. ijcai.org, 2019, pp. 3961–3967.

[99] H. Wu, C. Wang, Y. Tyshetskiy, A. Docherty, K. Lu, and L. Zhu, "Adversarial examples for graph data: Deep insights into attack and defense," in *Proceedings of the International Joint Conference on Artificial Intelligence*. ijcai.org, 2019, pp. 4816–4823.

[100] J. Chen, Y. Wu, X. Xu, Y. Chen, H. Zheng, and Q. Xuan, "Fast gradient attack on network embedding," *CoRR*, vol. abs/1809.02797, 2018.

[101] R. Yang and T. Long, "Derivative-free optimization adversarial attacks for graph convolutional networks," *PeerJ Computer Science*, vol. 7, p. e693, 2021.

[102] W. Jin, Y. Li, H. Xu, Y. Wang, and J. Tang, "Adversarial attacks and defenses on graphs: A review and empirical study," *CoRR*, vol. abs/2003.00653, 2020.

[103] J. Mu, B. Wang, Q. Li, K. Sun, M. Xu, and Z. Liu, "A hard label black-box adversarial attack against graph neural networks," *CoRR*, vol. abs/2108.09513, 2021.

[104] Q. Zheng, X. Zou, Y. Dong, Y. Cen, D. Yin, J. Xu, Y. Yang, and J. Tang, "Graph robustness benchmark: Benchmarking the adversarial robustness of graph machine learning," *CoRR*, vol. abs/2111.04314, 2021.

[105] N. Entezari, S. A. Al-Sayouri, A. Darvishzadeh, and E. E. Papalexakis, "All you need is low (rank): Defending against adversarial attacks on graphs," in *Proceedings of the ACM International Conference on Web Search and Data Mining*. ACM, 2020, pp. 169–177.

[106] W. Jin, Y. Ma, X. Liu, X. Tang, S. Wang, and J. Tang, "Graph structure learning for robust graph neural networks," in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 2020, pp. 66–74.

[107] Z. Xu and H. Tong, "Graph sanitation with application to node classification," *CoRR*, vol. abs/2105.09384, 2021.

[108] D. Zhu, Z. Zhang, P. Cui, and W. Zhu, "Robust graph convolutional networks against adversarial attacks," in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 2019, pp. 1399–1407.

[109] X. Tang, Y. Li, Y. Sun, H. Yao, P. Mitra, and S. Wang, "Transferring robustness for graph neural network against poisoning attacks," in *Proceedings of the ACM International Conference on Web Search and Data Mining*. ACM, 2020, pp. 600–608.

[110] V. N. Ioannidis and G. B. Giannakis, "Defending graph convolutional networks against adversarial attacks," in *IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2020, pp. 8469–8473.

[111] X. Zhang and M. Zitnik, "Gnnguard: Defending graph neural networks against adversarial attacks," in *Advances in Neural Information Processing Systems*, 2020.

[112] F. Feng, X. He, J. Tang, and T. Chua, "Graph adversarial training: Dynamically regularizing based on graph structure," *IEEE Transactions on Knowledge and Data Engineering*, vol. 33, no. 6, pp. 2493–2504, 2021.

[113] H. Jin and X. Zhang, "Robust training of graph convolutional networks via latent perturbation," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, ser. Lecture Notes in Computer Science, vol. 12459. Springer, 2020, pp. 394–411.

[114] A. Bojchevski and S. Günnemann, "Certifiable robustness to graph perturbations," in *Advances in Neural Information Processing Systems*, 2019, pp. 8317–8328.

[115] D. Zügner and S. Günnemann, "Certifiable robustness and robust training for graph convolutional networks," in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 2019, pp. 246–256.

[116] S. Tao, H. Shen, Q. Cao, L. Hou, and X. Cheng, "Adversarial immunization for certifiable robustness on graphs," in *Proceedings of the ACM International Conference on Web Search and Data Mining*. ACM, 2021, pp. 698–706.

[117] Y. Zhang, S. Khan, and M. Coates, "Comparing and detecting adversarial attacks for graph deep learning," in *Representation Learning on Graphs and Manifolds Workshop*, 2019.

[118] L. Chen, J. Li, Q. Peng, Y. Liu, Z. Zheng, and C. Yang, "Understanding structural vulnerability in graph convolutional networks," in *Proceedings of the International Joint Conference on Artificial Intelligence*. ijcai.org, 2021, pp. 2249–2255.

[119] T. Weng, H. Zhang, P. Chen, J. Yi, D. Su, Y. Gao, C. Hsieh, and L. Daniel, "Evaluating the robustness of neural networks: An extreme value theory approach," in *International Conference on Learning Representations*. OpenReview.net, 2018.

[120] F. Yu, Z. Qin, C. Liu, L. Zhao, Y. Wang, and X. Chen, "Interpreting and evaluating neural network robustness," in *Proceedings of the International Joint Conference on Artificial Intelligence*. ijcai.org, 2019, pp. 4199–4205.

[121] S. Geisler, T. Schmidt, H. Sirin, D. Zügner, A. Bojchevski, and S. Günnemann, "Robustness of graph neural networks at scale," in *Advances in Neural Information Processing Systems*, vol. 34, 2021.

[122] V. N. Ioannidis, D. Berberidis, and G. B. Giannakis, "Graphsac: Detecting anomalies in large-scale graphs," *CoRR*, vol. abs/1910.09589, 2019.

[123] S. Geisler, D. Zügner, and S. Günnemann, "Reliable graph neural networks via robust aggregation," in *Advances in Neural Information Processing Systems*, 2020.

[124] A. Mirhoseini, A. Goldie, M. Yazgan, J. W. Jiang, E. Songhori, S. Wang, Y.-J. Lee, E. Johnson, O. Pathak, A. Nazi *et al.*, "A graph placement methodology for fast chip design," *Nature*, vol. 594, no. 7862, pp. 207–212, 2021.

[125] T. Xia and W. Ku, "Geometric graph representation learning on protein structure prediction," in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 2021, pp. 1873–1883.

[126] Y. Wu, X. Wang, A. Zhang, X. He, and T. Chua, "Discovering invariant rationales for graph neural networks," in *International Conference on Learning Representations*, 2022.

[127] C. Abrate and F. Bonchi, "Counterfactual graphs for explainable classification of brain networks," in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2021.

[128] W. Fan, W. Jin, X. Liu, H. Xu, X. Tang, S. Wang, Q. Li, J. Tang, J. Wang, and C. C. Aggarwal, "Jointly attacking graph neural network and its explanations," *CoRR*, vol. abs/2108.03388, 2021.

[129] N. Liu, Q. Feng, and X. Hu, "Interpretability in graph neural networks," in *Graph Neural Networks: Foundations, Frontiers, and Applications*, L. Wu, P. Cui, J. Pei, and L. Zhao, Eds. Singapore: Springer Singapore, 2022, pp. 121–147.

[130] C. Molnar, *Interpretable machine learning*. Lulu. com, 2020.

[131] R. Milo, S. Shen-Orr, S. Itzkovitz, N. Kashtan, D. Chklovskii, and U. Alon, "Network motifs: simple building blocks of complex networks," *Science*, vol. 298, no. 5594, pp. 824–827, 2002.

[132] H. Yuan, J. Tang, X. Hu, and S. Ji, "XGNN: towards model-level explanations of graph neural networks," in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 2020, pp. 430–438.

[133] X. Zhao, B. Zong, Z. Guan, K. Zhang, and W. Zhao, "Substructure assembling network for graph classification," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2018.

[134] L. Chu, X. Hu, J. Hu, L. Wang, and J. Pei, "Exact and consistent interpretation for piecewise linear neural networks: A closed form solution," in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 2018, pp. 1244–1253.

[135] Q. Huang, M. Yamada, Y. Tian, D. Singh, D. Yin, and Y. Chang, "Graphlime: Local interpretable model explanations for graph neural networks," *CoRR*, vol. abs/2001.06216, 2020.

[136] M. Bajaj, L. Chu, Z. Y. Xue, J. Pei, L. Wang, P. C. Lam, and Y. Zhang, "Robust counterfactual explanations on graph neural networks," in *Advances in Neural Information Processing Systems*, 2021, pp. 5644–5655.

[137] C. Olah, A. Mordvintsev, and L. Schubert, "Feature visualization," *Distill*, 2017, https://distill.pub/2017/feature-visualization.

[138] J. Chen, L. Song, M. J. Wainwright, and M. I. Jordan, "Learning to explain: An information-theoretic perspective on model interpretation," in *Proceedings of the International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, vol. 80. PMLR, 2018, pp. 882–891.

[139] D. Duvenaud, D. Maclaurin, J. Aguilera-Iparraguirre, R. Gómez-Bombarelli, T. Hirzel, A. Aspuru-Guzik, and R. P. Adams, "Convolutional networks on graphs for learning molecular fingerprints," in *Advances in Neural Information Processing Systems*, 2015, pp. 2224–2232.

[140] Z. Liu, D. Zhou, and J. He, "Towards explainable representation of time-evolving graphs via spatial-temporal graph attention networks," in *Proceedings of the ACM International Conference on Information & Knowledge Management*. ACM, 2019, pp. 2137–2140.

[141] E. Dai and S. Wang, "Towards self-explainable graph neural network," in *Proceedings of the ACM International Conference on Information & Knowledge Management*. ACM, 2021, pp. 302–311.

[142] Z. Zhang, Q. Liu, H. Wang, C. Lu, and C. Lee, "Protgnn: Towards self-explaining graph neural networks," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2022.

[143] A. Subramonian, "Motif-driven contrastive learning of graph representations," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2021.

[144] J. Yu, T. Xu, Y. Rong, Y. Bian, J. Huang, and R. He, "Graph information bottleneck for subgraph recognition," in *International Conference on Learning Representations*, 2021.

[145] F. Baldassarre and H. Azizpour, "Explainability techniques for graph convolutional networks," in *Proceedings of the International Conference on Machine Learning*, 2019.

[146] R. Henderson, D. Clevert, and F. Montanari, "Improving molecular graph neural network explainability with orthonormalization and induced sparsity," in *Proceedings of the International Conference on Machine Learning*, 2021.

[147] J. Kang and H. Tong, "N2N: network derivative mining," in *Proceedings of the ACM International Conference on Information & Knowledge Management*. ACM, 2019, pp. 861–870.

[148] Y. Wang, Y. Yao, H. Tong, F. Xu, and J. Lu, "Discerning edge influence for network embedding," in *Proceedings of the ACM International Conference on Information & Knowledge Management*. ACM, 2019, pp. 429–438.

[149] X. Wang, Y. Wu, A. Zhang, X. He, and T. seng Chua, "Causal screening to interpret graph neural networks," 2021. [Online]. Available: https://openreview.net/forum?id=nzKv5vxZfge

[150] M. N. Vu and M. T. Thai, "Pgm-explainer: Probabilistic graphical model explanations for graph neural networks," in *Advances in Neural Information Processing Systems*, 2020.

[151] T. Schnake, O. Eberle, J. Lederer, S. Nakajima, K. T. Schütt, K.-R. Müller, and G. Montavon, "Higher-order explanations of graph neural networks via relevant walks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

[152] C. Shan, Y. Shen, Y. Zhang, X. Li, and D. Li, "Reinforcement learning enhanced explainer for graph neural networks," in *Advances in Neural Information Processing Systems*, 2021.

[153] W. Lin, H. Lan, H. Wang, and B. Li, "Orphicx: A causality-inspired latent variable model for interpreting graph neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2022.

[154] Q. Feng, N. Liu, F. Yang, R. Tang, M. Du, and X. Hu, "DEGREE: Decomposition based explanation for graph neural networks," in *International Conference on Learning Representations*, 2022.

[155] W. Lin, H. Lan, and B. Li, "Generative causal explanations for graph neural networks," in *Proceedings of the International Conference on Machine Learning*, 2021.

[156] M. Sundararajan, A. Taly, and Q. Yan, "Axiomatic attribution for deep networks," in *Proceedings of the International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, vol. 70. PMLR, 2017, pp. 3319–3328.

[157] A. Shrikumar, P. Greenside, and A. Kundaje, "Learning important features through propagating activation differences," in *Proceedings of the International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, vol. 70. PMLR, 2017, pp. 3145–3153.

[158] P. Schwab and W. Karlen, "Cxplain: Causal explanations for model interpretation under uncertainty," in *Advances in Neural Information Processing Systems*, 2019, pp. 10 220–10 230.

[159] T. Funke, M. Khosla, and A. Anand, "Hard masking for explaining graph neural networks," 2021. [Online]. Available: https://openreview.net/forum?id=uDN8pRAdsoC

[160] M. S. Schlichtkrull, N. D. Cao, and I. Titov, "Interpreting graph neural networks for NLP with differentiable edge masking," in *International Conference on Learning Representations*, 2021.

[161] H. Yuan, H. Yu, J. Wang, K. Li, and S. Ji, "On explainability of graph neural networks via subgraph explorations," in *Proceedings of the International Conference on Machine Learning*, vol. 139. PMLR, 2021, pp. 12 241–12 252.

[162] A. Lucic, M. ter Hoeve, G. Tolomei, M. de Rijke, and F. Silvestri, "Cf-gnnexplainer: Counterfactual explanations for graph neural networks," in *Proceedings of the International Conference on Artificial Intelligence and Statistics*, 2021.

[163] X. Wang, Y. Wu, A. Zhang, X. He, and T. seng Chua, "Towards multi-grained explainability for graph neural networks," in *Advances in Neural Information Processing Systems*, 2021.

[164] J. Tan, S. Geng, Z. Fu, Y. Ge, S. Xu, Y. Li, and Y. Zhang, "Learning and evaluating graph neural network explanations based on counterfactual and factual reasoning," in *Proceedings of the ACM Web Conference*, 2022.

[165] Y. Zhang, D. DeFazio, and A. Ramesh, "Relex: A model-agnostic relational model explainer," in *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*. ACM, 2021, pp. 1042–1049.

[166] J. Adebayo, J. Gilmer, M. Muelly, I. J. Goodfellow, M. Hardt, and B. Kim, "Sanity checks for saliency maps," in *Advances in Neural Information Processing Systems*, 2018.

[167] P. Dabkowski and Y. Gal, "Real time image saliency for black box classifiers," in *Advances in Neural Information Processing Systems*, 2017, pp. 6967–6976.

[168] J. Yao, S. Zhang, Y. Yao, F. Wang, J. Ma, J. Zhang, Y. Chu, L. Ji, K. Jia, T. Shen, A. Wu, F. Zhang, Z. Tan, K. Kuang, C. Wu, F. Wu, J. Zhou, and H. Yang, "Edge-cloud polarization and collaboration: A comprehensive survey," *CoRR*, vol. abs/2111.06061, 2021.

[169] M. A. Weinzierl and S. M. Harabagiu, "Automatic detection of COVID-19 vaccine misinformation with graph link prediction," *J. Biomed. Informatics*, vol. 124, p. 103955, 2021.

[170] X. Niu, B. Li, C. Li, R. Xiao, H. Sun, H. Deng, and Z. Chen, "A dual heterogeneous graph attention network to improve long-tail performance for shop search in e-commerce," in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 2020, pp. 3405–3415.

[171] Q. Deng, K. Wang, M. Zhao, Z. Zou, R. Wu, J. Tao, C. Fan, and L. Chen, "Personalized bundle recommendation in online games," in *Proceedings of the ACM International Conference on Information & Knowledge Management*. ACM, 2020, pp. 2381–2388.

[172] J. Shang, T. Ma, C. Xiao, and J. Sun, "Pre-training of graph augmented transformers for medication recommendation," in *Proceedings of the International Joint Conference on Artificial Intelligence*. ijcai.org, 2019, pp. 5953–5959.

[173] Y. Du, P. Luo, X. Hong, T. Xu, Z. Zhang, C. Ren, Y. Zheng, and E. Chen, "Inheritance-guided hierarchical assignment for clinical automatic diagnosis," in *International Conference on Database Systems for Advanced Applications*, ser. Lecture Notes in Computer Science, vol. 12683. Springer, 2021, pp. 461–477.

[174] V. Duddu, A. Boutet, and V. Shejwalkar, "Quantifying privacy leakage in graph embedding," in *Mobiquitous EAI International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services*. ACM, 2020, pp. 76–85.

[175] Z. Zhang, M. Chen, M. Backes, Y. Shen, and Y. Zhang, "Inference attacks against graph neural networks," in *USENIX Security Symposium*. USENIX Association, 2022.

[176] L. Franceschi, M. Niepert, M. Pontil, and X. He, "Learning discrete structures for graph neural networks," in *Proceedings of the International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, vol. 97. PMLR, 2019, pp. 1972–1982.

[177] Y. Wang and L. Sun, "Membership inference attacks on knowledge graphs," *CoRR*, vol. abs/2104.08273, 2021.

[178] Z. Zhang, Q. Liu, Z. Huang, H. Wang, C. Lu, C. Liu, and E. Chen, "Graphmi: Extracting private graph data from graph neural networks," in *Proceedings of the International Joint Conference on Artificial Intelligence*. ijcai.org, 2021, pp. 3749–3755.

[179] F. Wu, Y. Long, C. Zhang, and B. Li, "Linkteller: Recovering private edges from graph neural networks via influence analysis," in *IEEE Symposium on Security and Privacy*. IEEE, 2022.

[180] P. Kairouz, H. B. McMahan, B. Avent, A. Bellet, M. Bennis, A. N. Bhagoji, K. A. Bonawitz, Z. Charles, G. Cormode, R. Cummings, R. G. L. D'Oliveira, H. Eichner, S. E. Rouayheb, D. Evans, J. Gardner, Z. Garrett, A. Gascón, B. Ghazi, P. B. Gibbons, M. Gruteser, Z. Harchaoui, C. He, L. He, Z. Huo, B. Hutchinson, J. Hsu, M. Jaggi, T. Javidi, G. Joshi, M. Khodak, J. Konečný, A. Korolova, F. Koushanfar, S. Koyejo, T. Lepoint, Y. Liu, P. Mittal, M. Mohri, R. Nock, A. Özgür, R. Pagh, H. Qi, D. Ramage, R. Raskar, M. Raykova, D. Song, W. Song, S. U. Stich, Z. Sun, A. T. Suresh, F. Tramèr, P. Vepakomma, J. Wang, L. Xiong, Z. Xu, Q. Yang, F. X. Yu, H. Yu, and S. Zhao, "Advances and open problems in federated learning," *Foundations and Trends® in Machine Learning*, vol. 14, no. 1-2, pp. 1–210, 2021.

[181] H. Zhang, T. Shen, F. Wu, M. Yin, H. Yang, and C. Wu, "Federated graph learning - A position paper," *CoRR*, vol. abs/2105.11099, 2021.

[182] R. Kojima, S. Ishida, M. Ohta, H. Iwata, T. Honma, and Y. Okuno, "kgcn: a graph-based deep learning framework for chemical structures," *Journal of Cheminformatics*, vol. 12, no. 1, p. 32, 2020.

[183] R. Liu and H. Yu, "Federated graph neural networks: Overview, techniques and challenges," *CoRR*, vol. abs/2202.07256, 2022.

[184] Q. Yang, Y. Liu, T. Chen, and Y. Tong, "Federated machine learning: Concept and applications," *ACM Transactions on Intelligent Systems and Technology*, vol. 10, no. 2, pp. 12:1–12:19, 2019.

[185] B. Wang, A. Li, H. Li, and Y. Chen, "Graphfl: A federated learning framework for semi-supervised node classification on graphs," *CoRR*, vol. abs/2012.04187, 2020.

[186] L. Zheng, J. Zhou, C. Chen, B. Wu, L. Wang, and B. Zhang, "ASFGNN: automated separated-federated graph neural network," *Peer-to-Peer Networking and Applications*, vol. 14, no. 3, pp. 1692–1704, 2021.

[187] S. Yang, Z. Zhang, J. Zhou, Y. Wang, W. Sun, X. Zhong, Y. Fang, Q. Yu, and Y. Qi, "Financial risk analysis for smes with graph-based supply chain mining," in *Proceedings of the International Joint Conference on Artificial Intelligence*. ijcai.org, 2020, pp. 4661–4667.

[188] "Banca online webank: conto corrente online e mobile banking," https://www.webank.it/webankpub/wbresp/home.do, 2022, accessed April 19, 2022.

[189] "National vat invoice verification platform," https://inv-veri.chinatax.gov.cn/, 2022, accessed April 13, 2022.

[190] L. Ku, "Tencent's webank applying "federated learning" in a.i. - digital finance," 2019. [Online]. Available: https://www.digfingroup.com/webank-clustar/

[191] M. Jiang, T. Jung, R. Karl, and T. Zhao, "Federated dynamic graph neural networks with secure aggregation for video-based distributed surveillance," *ACM Transactions on Intelligent Systems and Technology*, 2021.

[192] C. He, E. Ceyani, K. Balasubramanian, M. Annavaram, and S. Avestimehr, "Spreadgnn: Serverless multi-task federated learning for graph neural networks," *CoRR*, vol. abs/2106.02743, 2021.

[193] H. Xie, J. Ma, L. Xiong, and C. Yang, "Federated graph classification over non-iid graphs," *CoRR*, vol. abs/2106.13423, 2021.

[194] H. C. Bayram and I. Rekik, "A federated multigraph integration approach for connectional brain template learning," in *International Workshop on Multimodal Learning for Clinical Decision Support*, ser. Lecture Notes in Computer Science, vol. 13050. Springer, 2021, pp. 36–47.

[195] W. Zhu, A. White, and J. Luo, "Federated learning of molecular properties in a heterogeneous setting," *CoRR*, vol. abs/2109.07258, 2021.

[196] G. Lou, Y. Liu, T. Zhang, and J. X. Zheng, "STFL: A temporal-spatial federated learning framework for graph neural networks," *CoRR*, vol. abs/2111.06750, 2021.

[197] C. Wu, F. Wu, Y. Cao, Y. Huang, and X. Xie, "Fedgnn: Federated graph neural network for privacy-preserving recommendation," *CoRR*, vol. abs/2102.04925, 2021.

[198] Z. Liu, L. Yang, Z. Fan, H. Peng, and P. S. Yu, "Federated social recommendation with graph neural network," *CoRR*, vol. abs/2111.10778, 2021.

[199] X. Ni, X. Xu, L. Lyu, C. Meng, and W. Wang, "A vertical federated learning framework for graph convolutional network," *CoRR*, vol. abs/2106.11593, 2021.

[200] J. Zhou, C. Chen, L. Zheng, H. Wu, J. Wu, X. Zheng, B. Wu, Z. Liu, and L. Wang, "Vertically federated graph neural network for privacy-preserving node classification," *CoRR*, vol. abs/2005.11903, 2021.

[201] C. Chen, W. Hu, Z. Xu, and Z. Zheng, "Fedgl: Federated graph learning framework with global self-supervision," *CoRR*, vol. abs/2105.03170, 2021.

[202] C. Meng, S. Rambhatla, and Y. Liu, "Cross-node federated graph neural network for spatio-temporal data modeling," in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 2021, pp. 1202–1211.

[203] K. Zhang, C. Yang, X. Li, L. Sun, and S. Yiu, "Subgraph federated learning with missing neighbor generation," *CoRR*, vol. abs/2106.13430, 2021.

[204] F. Chen, P. Li, T. Miyazaki, and C. Wu, "Fedgraph: Federated graph learning with intelligent sampling," *IEEE Transactions on Parallel and Distributed Systems*, vol. 33, no. 8, pp. 1775–1786, 2022.

[205] C. Shan, H. Jiao, and J. Fu, "Towards representation identical privacy-preserving graph neural network via split learning," *CoRR*, vol. abs/2107.05917, 2021.

[206] Y. Zhao, M. Li, L. Lai, N. Suda, D. Civin, and V. Chandra, "Federated learning with non-iid data," *CoRR*, vol. abs/1806.00582, 2018.

[207] N. Phan, X. Wu, H. Hu, and D. Dou, "Adaptive laplace mechanism: Differential privacy preservation in deep learning," in *IEEE International Conference on Data Mining*. IEEE Computer Society, 2017, pp. 385–394.

[208] S. Sajadmanesh and D. Gatica-Perez, "Locally private graph neural networks," in *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2021, pp. 2130–2145.

[209] D. Xu, S. Yuan, X. Wu, and H. Phan, "DPNE: differentially private network embedding," in *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, ser. Lecture Notes in Computer Science, vol. 10938. Springer, 2018, pp. 235–246.

[210] S. Zhang, H. Yin, T. Chen, Z. Huang, L. Cui, and X. Zhang, "Graph embedding for recommendation against attribute inference attacks," in *Proceedings of the ACM Web Conference*. ACM / IW3C2, 2021, pp. 3002–3014.

[211] P. Liao, H. Zhao, K. Xu, T. S. Jaakkola, G. J. Gordon, S. Jegelka, and R. Salakhutdinov, "Information obfuscation of graph neural networks," in *Proceedings of the International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, vol. 139. PMLR, 2021, pp. 6600–6610.

[212] K. Li, G. Luo, Y. Ye, W. Li, S. Ji, and Z. Cai, "Adversarial privacy-preserving graph embedding against inference attack," *IEEE Internet Things J.*, vol. 8, no. 8, pp. 6904–6915, 2021.

[213] F. Tramèr and D. Boneh, "Slalom: Fast, verifiable and private execution of neural networks in trusted hardware," in *International Conference on Learning Representations*. OpenReview.net, 2019.

[214] F. Mo, A. S. Shamsabadi, K. Katevas, S. Demetriou, I. Leontiadis, A. Cavallaro, and H. Haddadi, "Darknetz: towards model privacy at the edge using trusted execution environments," in *Proceedings of the International Conference on Mobile Systems*. ACM, 2020, pp. 161–174.

[215] Y. Zhang, G. Bai, X. Li, C. Curtis, C. Chen, and R. K. L. Ko, "Privcoll: Practical privacy-preserving collaborative machine learning," in *European Symposium on Research in Computer Security*, ser. Lecture Notes in Computer Science, vol. 12308. Springer, 2020, pp. 399–418.

[216] Q. Lou, B. Feng, G. C. Fox, and L. Jiang, "Glyph: Fast and accurately training deep neural networks on encrypted data," in *Advances in Neural Information Processing Systems*, 2020.

[217] P. Mohassel and Y. Zhang, "Secureml: A system for scalable privacy-preserving machine learning," in *IEEE Symposium on Security and Privacy*. IEEE Computer Society, 2017, pp. 19–38.

[218] X. Liu, B. Wu, X. Yuan, and X. Yi, "Leia: A lightweight cryptographic neural network inference system at the edge," *IEEE Transactions on Information Forensics and Security*, vol. 17, pp. 237–252, 2022.

[219] Y. Zheng, S. Lai, Y. Liu, X. Yuan, X. Yi, and C. Wang, "Aggregation service for federated learning: An efficient, secure, and more resilient realization," *The IEEE Transactions on Dependable and Secure Computing*, 2022.

[220] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, "Federated learning: Challenges, methods, and future directions," *IEEE Signal Processing Magazine*, vol. 37, no. 3, pp. 50–60, 2020.

[221] L. Shen, X. Chen, J. Shi, Y. Dong, and B. Fang, "An efficient 3-party framework for privacy-preserving neural network inference," in *European Symposium on Research in Computer Security*, ser. Lecture Notes in Computer Science, vol. 12308. Springer, 2020, pp. 419–439.

[222] L. Zhu, Z. Liu, and S. Han, "Deep leakage from gradients," in *Advances in Neural Information Processing Systems*, 2019, pp. 14 747–14 756.

[223] H. Yin, A. Mallya, A. Vahdat, J. M. Alvarez, J. Kautz, and P. Molchanov, "See through gradients: Image batch recovery via gradinversion," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. Computer Vision Foundation / IEEE, 2021, pp. 16 337–16 346.

[224] J. Zhu and M. B. Blaschko, "R-GAP: recursive gradient attack on privacy," in *International Conference on Learning Representations*. OpenReview.net, 2021.

[225] N. A. Saxena, K. Huang, E. DeFilippis, G. Radanovic, D. C. Parkes, and Y. Liu, "How do fairness definitions fare?: Examining public attitudes towards algorithmic definitions of fairness," in *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*. ACM, 2019, pp. 99–106.

[226] T. A. Rahman, B. Surma, M. Backes, and Y. Zhang, "Fairwalk: Towards fair graph embedding," in *Proceedings of the International Joint Conference on Artificial Intelligence*. ijcai.org, 2019, pp. 3289–3295.

[227] M. J. Kusner, J. R. Loftus, C. Russell, and R. Silva, "Counterfactual fairness," in *Advances in Neural Information Processing Systems*, 2017, pp. 4066–4076.

[228] S. Verma and J. Rubin, "Fairness definitions explained," in *Proceedings of the International Workshop on Software Fairness*. ACM, 2018, pp. 1–7.

[229] C. Dwork, M. Hardt, T. Pitassi, O. Reingold, and R. S. Zemel, "Fairness through awareness," in *Proceedings of the Innovations in Theoretical Computer Science Conference*. ACM, 2012, pp. 214–226.

[230] J. Kang and H. Tong, "Fair graph mining," in *Proceedings of the ACM International Conference on Information & Knowledge Management*. ACM, 2021, pp. 4849–4852.

[231] F. Kamiran and I. Zliobaite, "Explainable and non-explainable discrimination in classification," in *Discrimination and Privacy in the Information Society*, ser. Studies in Applied Philosophy, Epistemology and Rational Ethics. Springer, 2013, vol. 3, pp. 155–170.

[232] J. Chen, N. Kallus, X. Mao, G. Svacha, and M. Udell, "Fairness under unawareness: Assessing disparity when protected class is unobserved," in *Proceedings of the Conference on Fairness, Accountability, and Transparency*. ACM, 2019, pp. 339–348.

[233] L. Zhang, Y. Wu, and X. Wu, "A causal framework for discovering and removing direct and indirect discrimination," in *Proceedings of the International Joint Conference on Artificial Intelligence*. ijcai.org, 2017, pp. 3929–3935.

[234] D. H. Wolpert and W. G. Macready, "No free lunch theorems for optimization," *IEEE Transactions on Evolutionary Computation*, vol. 1, no. 1, pp. 67–82, 1997.

[235] A. J. Bose and W. L. Hamilton, "Compositional fairness constraints for graph embeddings," in *Proceedings of the International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, vol. 97. PMLR, 2019, pp. 715–724.

[236] J. Bromley, I. Guyon, Y. LeCun, E. Säckinger, and R. Shah, "Signature verification using a siamese time delay neural network," in *Advances in Neural Information Processing Systems*. Morgan Kaufmann, 1993, pp. 737–744.

[237] Y. Dong, N. Liu, B. Jalaian, and J. Li, "EDITS: modeling and mitigating data bias for graph neural networks," in *Proceedings of the ACM Web Conference*, 2022.

[238] Y. Liu, S. Pan, Y. G. Wang, F. Xiong, L. Wang, and V. C. S. Lee, "Anomaly detection in dynamic graphs via transformer," *IEEE Transactions on Knowledge and Data Engineering*, 2021.

[239] X. Zheng, Y. Liu, S. Pan, M. Zhang, D. Jin, and P. S. Yu, "Graph neural networks for graphs with heterophily: A survey," *CoRR*, vol. abs/2202.07082, 2022.

[240] S. Corbett-Davies, E. Pierson, A. Feller, S. Goel, and A. Huq, "Algorithmic decision making and the cost of fairness," in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 2017, pp. 797–806.

[241] J. Kang, Y. Zhu, Y. Xia, J. Luo, and H. Tong, "Rawlsgcn: Towards rawlsian difference principle on graph convolutional network," in *Proceedings of the ACM Web Conference*, 2022.

[242] R. Sun, P. Qiu, Y. Lyu, D. Wang, J. Dong, and G. Qu, "Lightning: Striking the secure isolation on GPU clouds with transient hardware faults," *CoRR*, vol. abs/2112.03662, 2021.

[243] J. Feigenbaum, A. D. Jaggard, and R. N. Wright, "Accountability in computing: Concepts and mechanisms," *Foundations and Trends in Privacy and Security*, vol. 2, no. 4, pp. 247–399, 2020.

[244] E. Commision, "Europe fit for the digital age: Commission proposes new rules and actions for excellence and trust in artificial intelligence," *Europan Commision: Geneva, Switzerland*, 2021.

[245] Y. Yuan, W. Wang, G. M. Coghill, and W. Pang, "A novel genetic algorithm with hierarchical evaluation strategy for hyperparameter optimisation of graph neural networks," *CoRR*, vol. abs/2101.09300, 2021.

[246] Y. Wang, "Bag of tricks of semi-supervised classification with graph neural networks," *CoRR*, vol. abs/2103.13355, 2021.

[247] T. Chen, K. Zhou, K. Duan, W. Zheng, P. Wang, X. Hu, and Z. Wang, "Bag of tricks for training deeper graph neural networks: A comprehensive benchmark study," *CoRR*, vol. abs/2108.10521, 2021.

[248] F. Errica, M. Podda, D. Bacciu, and A. Micheli, "A fair comparison of graph neural networks for graph classification," in *International Conference on Learning Representations*. OpenReview.net, 2020.

[249] M. Javaheripi and F. Koushanfar, "HASHTAG: hash signatures for online detection of fault-injection attacks on deep neural networks," *CoRR*, vol. abs/2111.01932, 2021.

[250] Z. He, T. Zhang, and R. B. Lee, "Sensitive-sample fingerprinting of deep neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. Computer Vision Foundation / IEEE, 2019, pp. 4729–4737.

[251] H. Jia, M. Yaghini, C. A. Choquette-Choo, N. Dullerud, A. Thudi, V. Chandrasekaran, and N. Papernot, "Proof-of-learning: Definitions and practice," in *IEEE Symposium on Security and Privacy*. IEEE, 2021, pp. 1039–1056.

[252] Y. Lan, Y. Liu, B. Li, and C. Miao, "Proof of learning (pole): Empowering machine learning with consensus building on blockchains (demo)," in *Proceedings of the AAAI Conference on Artificial Intelligence*. AAAI Press, 2021, pp. 16 063–16 066.

[253] E. Strubell, A. Ganesh, and A. McCallum, "Energy and policy considerations for deep learning in NLP," in *Proceedings of the Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 2019, pp. 3645–3650.

[254] W. Hu, M. Fey, M. Zitnik, Y. Dong, H. Ren, B. Liu, M. Catasta, and J. Leskovec, "Open graph benchmark: Datasets for machine learning on graphs," in *Advances in Neural Information Processing Systems*, 2020.

[255] Y. Rong, W. Huang, T. Xu, and J. Huang, "Dropedge: Towards deep graph convolutional networks on node classification," in *International Conference on Learning Representations*. OpenReview.net, 2020.

[256] Y. Zhao, D. Wang, D. Bates, R. D. Mullins, M. Jamnik, and P. Liò, "Learned low precision graph neural networks," *CoRR*, vol. abs/2009.09232, 2020.

[257] B. Yan, C. Wang, G. Guo, and Y. Lou, "Tinygnn: Learning efficient graph neural networks," in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 2020, pp. 1848–1856.

[258] S. A. Tailor, J. Fernández-Marqués, and N. D. Lane, "Degree-quant: Quantization-aware training for graph neural networks," in *International Conference on Learning Representations*. OpenReview.net, 2021.

[259] H. Zhou, A. Srivastava, H. Zeng, R. Kannan, and V. K. Prasanna, "Accelerating large scale real-time GNN inference using channel pruning," *Proceedings of the VLDB Endowment*, vol. 14, no. 9, pp. 1597–1605, 2021.

[260] Y. Bai, C. Li, Z. Lin, Y. Wu, Y. Miao, Y. Liu, and Y. Xu, "Efficient data loader for fast sampling-based GNN training on large graphs," *IEEE Transactions on Parallel and Distributed Systems*, vol. 32, no. 10, pp. 2541–2556, 2021.

[261] Z. Cai, X. Yan, Y. Wu, K. Ma, J. Cheng, and F. Yu, "DGCL: an efficient communication library for distributed GNN training," in *Proceedings of the European Conference on Computer Systems*. ACM, 2021, pp. 130–144.

[262] M. Yan, L. Deng, X. Hu, L. Liang, Y. Feng, X. Ye, Z. Zhang, D. Fan, and Y. Xie, "Hygcn: A GCN accelerator with hybrid architecture," in *IEEE International Symposium on High Performance Computer Architecture*. IEEE, 2020, pp. 15–29.

[263] A. Zhou, J. Yang, Y. Gao, T. Qiao, Y. Qi, X. Wang, Y. Chen, P. Dai, W. Zhao, and C. Hu, "Brief industry paper: optimizing memory efficiency of graph neural networks on edge computing platforms," in *IEEE Real-Time and Embedded Technology and Applications Symposium*. IEEE, 2021, pp. 445–448.

[264] Z. Zhu, S. Xu, J. Tang, and M. Qu, "Graphvite: A high-performance CPU-GPU hybrid system for node embedding," in *Proceedings of the ACM Web Conference*. ACM, 2019, pp. 2494–2504.

[265] M. Fey, J. E. Lenssen, F. Weichert, and J. Leskovec, "Gnnautoscale: Scalable and expressive graph neural networks via historical embeddings," in *Proceedings of the International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, vol. 139. PMLR, 2021, pp. 3294–3304.

[266] A. Derrow-Pinion, J. She, D. Wong, O. Lange, T. Hester, L. Perez, M. Nunkesser, S. Lee, X. Guo, B. Wiltshire, P. W. Battaglia, V. Gupta, A. Li, Z. Xu, A. Sanchez-Gonzalez, Y. Li, and P. Velickovic, "ETA prediction with graph neural networks in google maps," in *Proceedings of the ACM International Conference on Information & Knowledge Management*. ACM, 2021, pp. 3767–3776.

[267] Y. Chen, Y. Bian, X. Xiao, Y. Rong, T. Xu, and J. Huang, "On self-distilling graph neural network," in *Proceedings of the International Joint Conference on Artificial Intelligence*. ijcai.org, 2021, pp. 2278–2284.

[268] X. Deng and Z. Zhang, "Graph-free knowledge distillation for graph neural networks," in *Proceedings of the International Joint Conference on Artificial Intelligence*. ijcai.org, 2021, pp. 2321–2327.

[269] T. Chen, Y. Sui, X. Chen, A. Zhang, and Z. Wang, "A unified lottery ticket hypothesis for graph neural networks," in *Proceedings of the International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, vol. 139. PMLR, 2021, pp. 1695–1706.

[270] M. Bahri, G. Bahl, and S. Zafeiriou, "Binary graph neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. Computer Vision Foundation / IEEE, 2021, pp. 9492–9501.

[271] M. Fey and J. E. Lenssen, "Fast graph representation learning with PyTorch Geometric," in *International Conference on Learning Representations*, 2019.

[272] M. Wang, L. Yu, D. Zheng, Q. Gan, Y. Gai, Z. Ye, M. Li, J. Zhou, Q. Huang, C. Ma, Z. Huang, Q. Guo, H. Zhang, H. Lin, J. Zhao, J. Li, A. J. Smola, and Z. Zhang, "Deep graph library: Towards efficient and scalable deep learning on graphs," *CoRR*, vol. abs/1909.01315, 2019.

[273] S. Liang, Y. Wang, C. Liu, L. He, H. Li, D. Xu, and X. Li, "Engn: A high-throughput and energy-efficient accelerator for large graph neural networks," *IEEE Transactions on Computers*, vol. 70, no. 9, pp. 1511–1525, 2021.

[274] M. Yan, Z. Chen, L. Deng, X. Ye, Z. Zhang, D. Fan, and Y. Xie, "Characterizing and understanding gcns on GPU," *IEEE Computer Architecture Letters*, vol. 19, no. 1, pp. 22–25, 2020.

[275] M. Yan, X. Hu, S. Li, A. Basak, H. Li, X. Ma, I. Akgun, Y. Feng, P. Gu, L. Deng, X. Ye, Z. Zhang, D. Fan, and Y. Xie, "Alleviating irregularity in graph analytics acceleration: a hardware/software co-design approach," in *Annual IEEE/ACM International Symposium on Microarchitecture*. ACM, 2019, pp. 615–628.

[276] Z. Xi, R. Pang, S. Ji, and T. Wang, "Graph backdoor," in *USENIX Security Symposium*. USENIX Association, 2021, pp. 1523–1540.

[277] H. Peng, H. Li, Y. Song, V. W. Zheng, and J. Li, "Differentially private federated knowledge graphs embedding," in *Proceedings of the ACM International Conference on Information & Knowledge Management*. ACM, 2021, pp. 1416–1425.

[278] H. Xie, J. Ma, L. Xiong, and C. Yang, "Federated graph classification over non-iid graphs," in *Advances in Neural Information Processing Systems*, 2021.

[279] D. Liu, A. Lamb, K. Kawaguchi, A. Goyal, C. Sun, M. C. Mozer, and Y. Bengio, "Discrete-valued neural communication," in *Advances in Neural Information Processing Systems*, 2021.

[280] A. Zhang and J. Ma, "Defensevgae: Defending against adversarial attacks on graph data via a variational graph autoencoder," *CoRR*, vol. abs/2006.08900, 2020.

[281] C. Yang, J. Liu, and C. Shi, "Extract the knowledge of graph neural networks and go beyond it: An effective knowledge distillation framework," in *Proceedings of the ACM Web Conference*. ACM / IW3C2, 2021, pp. 1227–1237.

[282] T. Lacombe, Y. Ike, M. Carrière, F. Chazal, M. Glisse, and Y. Umeda, "Topological uncertainty: Monitoring trained neural networks through persistence of activation graphs," in *Proceedings of the International Joint Conference on Artificial Intelligence*. ijcai.org, 2021, pp. 2666–2672.

[283] W. Chiang, X. Liu, S. Si, Y. Li, S. Bengio, and C. Hsieh, "Cluster-gcn: An efficient algorithm for training deep and large graph convolutional networks," in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 2019, pp. 257–266.

[284] H. Zeng, H. Zhou, A. Srivastava, R. Kannan, and V. K. Prasanna, "Accurate, efficient and scalable graph embedding," in *IEEE International Parallel and Distributed Processing Symposium*. IEEE, 2019, pp. 462–471.

[285] Y. Gao, H. Yang, P. Zhang, C. Zhou, and Y. Hu, "Graph neural architecture search," in *Proceedings of the International Joint Conference on Artificial Intelligence*. ijcai.org, 2020, pp. 1403–1409.

[286] Y. Yang, J. Qiu, M. Song, D. Tao, and X. Wang, "Distilling knowledge from graph convolutional networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. Computer Vision Foundation / IEEE, 2020, pp. 7072–7081.

[287] D. Stamoulis, T. R. Chin, A. K. Prakash, H. Fang, S. Sajja, M. Bognar, and D. Marculescu, "Designing adaptive neural networks for energy-constrained image classification," in *IEEE/ACM International Conference on Computer-Aided Design*. ACM, 2018, p. 23.

[288] Z. Zhang, X. Wang, and W. Zhu, "Automated machine learning on graphs: A survey," in *Proceedings of the International Joint Conference on Artificial Intelligence*, 2021.

[289] Y. Zhang, H. You, Y. Fu, T. Geng, A. Li, and Y. Lin, "G-cos: Gnn-accelerator co-search towards both better accuracy and efficiency," in *IEEE/ACM International Conference on Computer-Aided Design*, 2021.

[290] H. Yang, L. Duan, Y. Chen, and H. Li, "BSQ: exploring bit-level sparsity for mixed-precision neural network quantization," in *International Conference on Learning Representations*. OpenReview.net, 2021.

[291] Y. Liu, Y. Zheng, D. Zhang, H. Chen, H. Peng, and S. Pan, "Towards unsupervised deep graph structure learning," in *TheWebConf (WWW)*. ACM, 2022, pp. 1392–1403.

[292] J. Rawls, "A theory of justice," in *A theory of justice*. Harvard university press, 2020.

[293] "Trusted ai," https://research.ibm.com/teams/trusted-ai#tools, accessed December 30, 2021.

[294] C. Morris, N. M. Kriege, F. Bause, K. Kersting, P. Mutzel, and M. Neumann, "Tudataset: A collection of benchmark datasets for learning with graphs," in *Proceedings of the International Conference on Machine Learning Workshop on Graph Representation Learning and Beyond*, 2020. [Online]. Available: www.graphlearning.io

[295] X. Wang, H. Liu, C. Shi, and C. Yang, "Be confident! towards trustworthy graph neural networks via confidence calibration," in *Advances in Neural Information Processing Systems*, 2021.

[296] M. Wang, D. Zheng, Z. Ye, Q. Gan, M. Li, X. Song, J. Zhou, C. Ma, L. Yu, Y. Gai, T. Xiao, T. He, G. Karypis, J. Li, and Z. Zhang, "Deep graph library: A graph-centric, highly-performant package for graph neural networks," *CoRR*, vol. abs/1909.01315, 2019.

[297] J. Li, K. Xu, L. Chen, Z. Zheng, and X. Liu, "Graphgallery: A platform for fast benchmarking and easy development of graph neural networks based intelligent software," in *IEEE/ACM International Conference on Software Engineering: Companion Proceedings*. IEEE, 2021, pp. 13–16.

[298] W. Hu, M. Fey, H. Ren, M. Nakata, Y. Dong, and J. Leskovec, "OGB-LSC: A large-scale challenge for machine learning on graphs," *CoRR*, vol. abs/2103.09430, 2021.

[299] F. Masrour, T. Wilson, H. Yan, P. Tan, and A. Esfahanian, "Bursting the filter bubble: Fairness-aware network link prediction," in *Proceedings of the AAAI Conference on Artificial Intelligence*. AAAI Press, 2020, pp. 841–848.