Machine Intelligence Research
www.mi-research.net

22(1), February 2025, 17-59
DOI: 10.1007/s11633-024-1519-z

# Counterfactual Learning on Graphs: A Survey

Zhimeng Guo[1] Zongyu Wu[1] Teng Xiao[1] Charu Aggarwal[2]
Hui Liu[3] Suhang Wang[1]

[1] College of Information Sciences and Technology, Pennsylvania State University, University Park 16802, USA

[2] International Business Machines Corporation T.J. Watson Research Center, New York 10598, USA

[3] College of Engineering, Michigan State University, East Lansing 48824, USA

**Abstract:** Graph-structured data are pervasive in the real-world such as social networks, molecular graphs and transaction networks. Graph neural networks (GNNs) have achieved great success in representation learning on graphs, facilitating various downstream tasks. However, GNNs have several drawbacks such as lacking interpretability, can easily inherit the bias of data and cannot model casual relations. Recently, counterfactual learning on graphs has shown promising results in alleviating these drawbacks. Various approaches have been proposed for counterfactual fairness, explainability, link prediction and other applications on graphs. To facilitate the development of this promising direction, in this survey, we categorize and comprehensively review papers on graph counterfactual learning. We divide existing methods into four categories based on problems studied. For each category, we provide background and motivating examples, a general framework summarizing existing works and a detailed review of these works. We point out promising future research directions at the intersection of graph-structured data, counterfactual learning, and real-world applications. To offer a comprehensive view of resources for future studies, we compile a collection of open-source implementations, public datasets, and commonly-used evaluation metrics. This survey aims to serve as a "one-stop-shop" for building a unified understanding of graph counterfactual learning categories and current resources.

**Keywords:** Counterfactual learning, graph-structured data, graph neural networks, fairness, explainability.

**Citation:** Z. Guo, Z. Wu, T. Xiao, C. Aggarwal, H. Liu, S. Wang. Counterfactual learning on graphs: A survey. *Machine Intelligence Research*, vol.22, no.1, pp.17–59, 2025. http://doi.org/10.1007/s11633-024-1519-z

## 1 Introduction

Graphs are a ubiquitous data structure and a universal language for representing objects and complex interactions[1]. They can model a wide range of real-world systems, such as social networks[2], chemical compounds[3], knowledge graphs[4], and recommendation systems[5]. For instance, in social networks[6], nodes represent people, and edges between nodes denote social connections between them. In molecular graphs, nodes correspond to atoms, and edges represent the chemical bonds between them, providing a structural representation of chemical compounds that can be used for tasks like drug discovery or material design[7, 8]. The pervasiveness of graph-structured data has raised the broad attention of researchers on graph analytics and mining, and various methods have been proposed[9].

Network representation[10, 11], which aims to learn low-dimensional vector representations of nodes or graphs that capture the intrinsic feature and structure information of nodes or graphs, is one essential task of graph mining. The learned representation can facilitate various downstream tasks such as node classification[12], link prediction[13], community detection[14] and graph classification[15]. Neural networks have shown great power in representation learning for many domains such as computer vision[16], natural language processing[17], etc. Neural network-based methods have also inspired the emergence and flourishing of graph neural networks (GNNs)[12, 18, 19]. Since graph convolutional network[12] was proposed, there have been various variants of GNNs[20–22]. They greatly boost the development of graph learning methods and have achieved state-of-the-art performance on many graph mining tasks, e.g., node classification[22], link prediction[23] and graph classification[24]. Due to the great power of graph learning methods, they have been successfully applied in many high-stakes decision scenarios, such as drug discovery[25], fake news detection[26] and financial analysis[27].

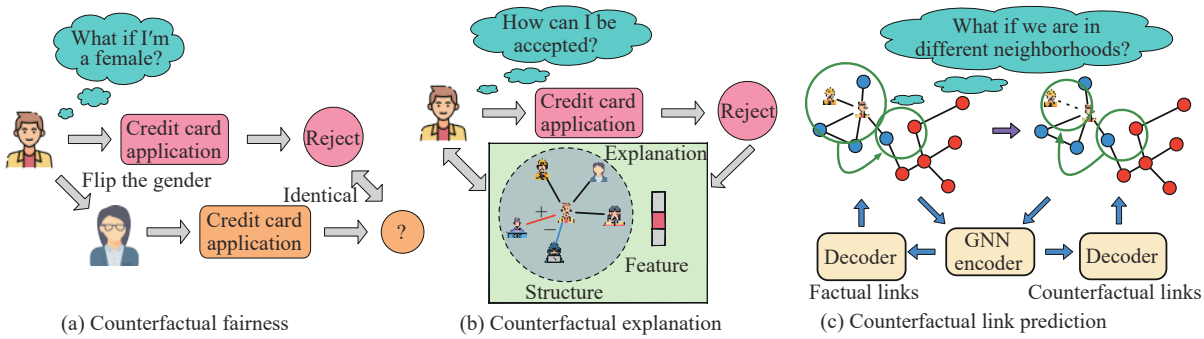
Despite the great potential of real-world benefits, recent studies show that existing graph learning methods tend to inherit the bias pattern from the biased dataset[28], lack the interpretability[29] and cannot exploit the rich information stored in graph data[30]. For example, with the biased dataset, GNNs are easy to learn an unfair classifier, e.g., give applicants different decisions

Review
Manuscript received on April 10, 2024; accepted on July 8, 2024
Recommended by Associate Editor Liwei Wang
Colored figures are available in the online version at https://link.springer.com/journal/11633

based on their races or other sensitive information[31, 32]. These issues severely hinder trust in the model and limit the real-world application of graph learning methods[33]. Counterfactual learning gives a chance to alleviate the intrinsic bias[34], making models interpretable[35] and exploiting the information stored in data well[36]. The notation of counterfactual comes from the research community of causal inference[37]. Counterfactual reasoning aims to answer "what would have happened, given the knowledge of what in fact happened". The ability to learn with counterfactuals and generalize to unseen environments is considered a significant component of general AI. The topic of learning causality has been well studied in many areas, such as economics[38], education[39], and medical science[40]. To know the causal effect of an action, we need to know the factual outcome with the observed action and the counterfactual outcome with unobserved action. A straightforward approach is to conduct randomized controlled trials to get counterfactual outcomes. However, in the real-world setting, we only have access to the observational factual data, i.e., the observed action and its corresponding factual outcome, which is a key challenge to learn causality[41]. Fortunately, the development of information technology gives abundant data sources that we can take advantage of to find the implied information in the data[42]. Hence, the core question is how to get the counterfactuals from the observational data[43], and how to use the counterfactuals to aid machine learning models[42].

Counterfactual learning on graphs is an emerging direction and only has a very short history[42]. However, recent works on graph counterfactual learning have shown great potential to overcome the aforementioned challenges on fairness[44], explanation[45], etc. In Fig. 1, we show some motivation examples for graph counterfactual learning. Concretely, equipped with counterfactual learning, we can go beyond the fairness definition at the group level and achieve fairness for each individual as in the factual world and the counterfactual world[34], where the individual belongs to a different demographic group. As depicted in Fig. 1(a), the goal of counterfactual fairness is to ensure that an applicant and his counterfactual counterpart (with a different gender) receive the same credit card application outcome. For counterfactual explanation on graphs, in addition to finding a compact subgraph which is highly correlated to the prediction[46], it aims at finding a reasonable change to have different result[47], which can be used to not only answer why the model gives such prediction but also give suggestions on what to do in order to achieve another desired result. As illustrated in Fig. 1(b), in a credit card application scenario[35], when an applicant is rejected, a conventional explanation might state that their "credit score was too low." In contrast, a counterfactual explanation could provide actionable recommendations on what minimal changes (e.g., in transaction relationships) the customer could make to alter the decision and ultimately gain approval. Besides the aid on fairness and interpretability, the research community also utilizes counterfactual learning to provide additional information from the counterfactual world, e.g., using both factual links and counterfactual links to help build more powerful GNNs[30]. As shown in Fig. 1(c), two friends live in the same neighborhood. By placing them in different neighborhoods, GNNs can infer the counterfactual link between them. This enables GNNs to gain a deeper understanding of the causal factors that shape their relationships while mitigating the impact of neighborhood factors. Considering the increasing trend of graph counterfactual learning and the diversity of related pretext tasks, there is an urgent need to have a systematic taxonomy to summarize the methodologies and applications of graph counterfactual learning.

To fill the gap, this survey paper conducts the first comprehensive and up-to-date overview of the booming area of graph counterfactual learning, provides some insights and potential future directions, and creates a "one-step-stop" that collects a set of open-source implementations, public datasets and commonly-used evaluation metrics together. The intended audiences for this article are general machine learning researchers who would like to know graph counterfactual learning, graph learning researchers who want to keep track of the most recent advances in graph neural networks, and domain experts who would like to generalize graph counterfactual learn-




(a) Counterfactual fairness (b) Counterfactual explanation (c) Counterfactual link prediction

Fig. 1 An illustration of counterfactual learning on graphs (Colored figures are available in the online version at https://link.springer.com/journal/11633)

ing to new applications or other fields. The core contributions of this survey are:

**1) The first survey of counterfactual learning on graphs**. To the best of our knowledge, our survey is the first to review counterfactual learning techniques for graphs. The most relevant surveys are about causal inference[37, 41] and causal machine learning[42]. At the time the survey was initially completed, there has been no dedicated and comprehensive survey about causal learning in graph domain.

**2) A comprehensive and up-to-date review.** We review the most up-to-date graph counterfactual learning techniques published in influential international conferences and journals of deep learning, data mining, computer vision, natural language processing, and artificial intelligence, including ACM TOIS, ICLR, NeurIPS, ICML, SIGKDD, WSDM, CIKM, WWW, ICDM, NAACL, IJCAI, AAAI and others. We also include papers in other domains like chemical science.

**3) Systematic taxonomy and unified frameworks.** We systematically categorize existing works into counterfactual fairness, counterfactual explanation, counterfactual link prediction and recommendation, and applications. For most of the categories, we provide unified frameworks that mathematically formalize graph counterfactual learning approaches in each category. An overview of the taxonomy is shown in Fig. 2.

**4) Future directions and "one-step-stop" for resources.** From the survey results, we point out promising and important future directions. We also provide a collection of open-source implementations, public datasets, and commonly used evaluation metrics to facilitate the community. We maintain a repository containing papers in graph counterfactual learning and we will keep updating these papers in the repository: https://github.com/TimeLovercc/Awesome-Graph-Causal-Learning.

Comparison is with related survey articles. Table 1 highlights the differences between our survey and related survey papers. Most existing surveys primarily focus on general causal inference[37, 41], counterfactual fairness[34], and counterfactual explanation[35], seldom discussing research progress on graph data. While other graph domain surveys address fairness[33] and interpretability[33, 58], they rarely summarize existing work from causal or counterfactual learning perspectives[52]. Our survey provides the first comprehensive overview of graph counterfactual learning, offering causal learning background, reviewing graph counterfactual learning techniques for fairness-aware models, explainable models, link prediction, recommender systems, real-world applications, and promising research directions. Hence, our survey is distinct from existing surveys and can support the growth of this important and emerging domain.

The overview of this survey is shown in Fig. 2. Section 2 defines the related concepts and gives notations which will be used in the following sections. Sections 3 and 4 describe the unified framework of counterfactual fairness and counterfactual explanation on graph data, respectively. We also summarize useful resources including evaluation metrics and datasets. Section 5 reviews the

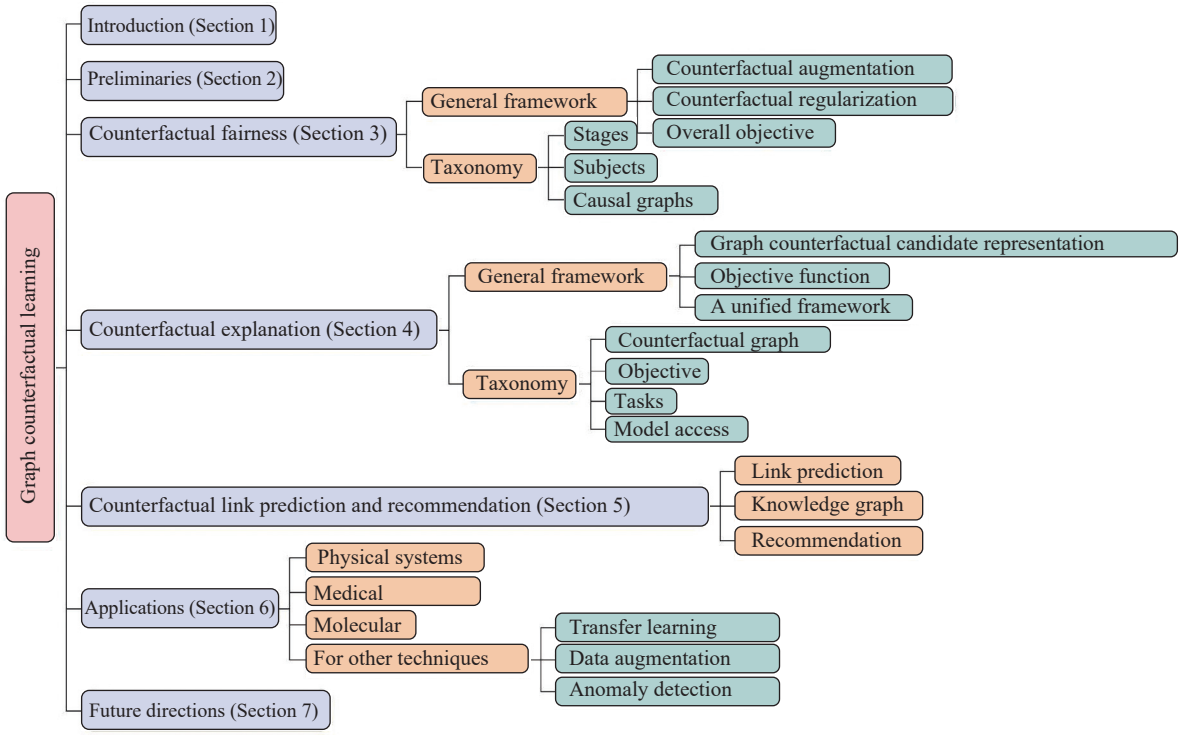



Fig. 2 Overview of graph counterfactual learning

Table 1 A comparison between existing surveys on causal inference and graph learning and ours. Unified denotes unified framework.

| Surveys | Graph | Unified | Causal | Counterfactual | | | Applications | Source code | Dataset | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | Fairness | Explanation | Data augmentation | | | Real-world | Synthetic |
| Our survey | • | • | • | • | • | • | • | • | • | • |
| Dai and Wang[29] | • | • | – | • | – | – | • | – | • | • |
| Mehrabi et al.[44] | – | – | • | • | – | – | • | – | • | – |
| Makhlouf et al.[48] | – | • | • | • | – | – | – | • | – | |
| Yao et al.[37] | – | • | • | – | – | • | • | • | • | • |
| Kaddour et al.[42] | • | – | • | • | • | • | • | – | – | – |
| Guo et al.[41] | – | • | • | – | – | • | • | – | • | • |
| Verma et al.[35] | – | • | • | – | • | – | – | – | • | – |
| Stepin et al.[49] | – | – | • | – | • | – | – | – | – | – |
| Artelt and Hammer[50] | – | • | – | – | • | – | – | | – | – |
| Oneto et al.[51] | • | – | – | • | • | – | – | – | – | – |
| Prado-Romero et al.[52] | • | • | • | – | • | – | • | – | • | • |
| Feder et al.[53] | – | • | • | • | • | – | – | – | – | • |
| Cheng et al.[54] | – | – | • | • | • | • | – | – | – | – |
| Liu et al.[55] | – | – | • | – | – | • | • | – | • | • |
| Sanchez et al.[56] | – | – | • | – | – | • | • | – | – | – |
| Vlontzos et al.[57] | – | – | • | • | • | – | – | – | – | – |

categories of counterfactual link prediction and counterfactual recommendation. Section 6 surveys the real-world applications of graph counterfactual learning in various domains. Section 7 points out the unsolved challenges and promising future directions. Section 8 concludes this survey.

## 2 Preliminary

In this section, we give the notations and definitions frequently used in this survey. We also provide some background knowledge for the following sections.

### 2.1 Notations and definitions

Throughout the paper, variables are denoted by capital letters. Lowercase letters denote specific values of variables. Matrices are written in boldface capital letters and vectors are denoted in boldface lowercase letters, respectively. Capital letters in calligraphic math font are used to denote sets. Such as $\mathcal{P}$ is used to denote sets and $|\mathcal{P}|$ denotes the cardinality of $\mathcal{P}$. Let $\mathcal{G} = \{\mathcal{V}, \mathcal{E}, \mathbf{X}\}$ represent a graph, where $\mathcal{V} = \{v_1, \cdots, v_N\}$ denotes the set of $N$ nodes, $\mathcal{E}$ represents the set of edges, and $\mathbf{X} \in \mathbf{R}^{N \times d_x}$ signifies the node features. The $i$-th row of $\mathbf{X}$, $\mathbf{x}_i$, is a $d_x$-dimensional feature vector for node $v_i$. The graph's adjacency matrix is denoted as $\mathbf{A} \in \{0,1\}^{n \times n}$, where $\mathbf{A}_{ij} = 1$ if there is an edge from $v_i$ to $v_j$; otherwise, $\mathbf{A}_{ij} = 0$. The labels of graphs or nodes, depending on the task, are represented by $\mathbf{Y} = \{y_1, \cdots, y_N\}$, where $y_i \in \{1, 2, \cdots, C\}$ is the label of the $i$-th data sample and $C$ denotes the number of classes. Table 2 summarizes the most frequently used notations and definitions. Next, we provide basic definitions for causal inference and counterfactual learning within the potential outcome framework[37, 59, 60].

Table 2 Notations and explanations

| Notations | Explanations |
|---|---|
| $d$ | Dimension of vectors |
| $N$ | Number of nodes |
| $\mathbf{A}$ | Adjacency matrix |
| $\mathbf{x}_i$ or $\mathbf{x}_i^{\mathrm{F}}$ | Factual attributes of node $v_i$ |
| $\mathbf{x}_i^{\mathrm{CF}}$ | Counterfactual attributes of node $v_i$ |
| $\mathbf{h}_i$ or $\mathbf{h}_i^{\mathrm{F}}$ | Factual embeddings of node $v_i$ |
| $\mathbf{h}_i^{\mathrm{CF}}$ | Counterfactual embeddings of node $v_i$ |
| $\mathbf{Y}$ | Labels |
| $\mathcal{N}_i$ | Neighbors of node $v_i$ |
| $\sigma$ | Sigmoid function |
| $\odot$ | Hadamard product |
| $S$ and $s$ | Sensitive variable and sensitive attribute |

**Definition 1** (unit[37]). A unit is an atomic research object when studying the treatment effect.

**Definition 2** (treatment). Treatment refers to the action that applies (exposes or subjects) to a unit. We consider binary treatment (0/1) here. We use $W$ and $w$ to

denote the random variable of treatment and the instance of treatment, respectively. For the observation treatment $W = w$, $w'$ stands for the counterfactual treatment, i.e., $w' = 1 - w$.

**Definition 3** (potential outcome). For each individual, there exist two potential outcomes denoted as $Y(0)$ and $Y(1) \in \mathcal{C}$, where $Y(0)$ is a potential outcome associated with treatment $W = 0$ and $Y(1)$ is the potential outcome associated with $W = 1$. Note that each individual receives only one treatment and reveals the factual outcome value for the received treatment.

**Definition 4** (observed outcome (factual outcome) and counterfactual outcome[37]). The observed outcome $Y(W = w)$ is the outcome when applying the actual treatment $w$. Counterfactual outcome $Y(W = w')$ is the outcome if the unit had taken treatment $w'$ given that the unit actually took treatment $w$.

## 2.2 Graph learning

Graphs are a common data structure used in various fields like social networks[2], brain connectomes[61], trade networks[1], and recommender systems[62]. Graph learning tasks are divided into graph-level, node-level and edge-level tasks[12, 23, 63]. Both node features and geometric structures are crucial for these tasks[1, 12], so effectively encoding graph structure and node attributes into vector representations is essential. Simple approaches extract basic graph properties[64], while others use graph statistics like kernel functions[65], local neighborhood structure[66], and graphlet orbits[67]. However, these methods may be limited in flexibility, expressiveness, and computational efficiency due to their reliance on hand-engineered features[1].

Another line of research focuses on network representation learning, aiming to learn low-dimensional vector representations of nodes or graphs[10–12, 68, 69]. Among these, graph neural networks (GNNs)[12, 19, 20, 69–74] have demonstrated a strong ability to learn node and graph representations, facilitating various downstream tasks[12, 19, 20, 69, 75–78]. GNNs generally adopt a message-passing mechanism that learns a node representation by iteratively aggregating the node's neighborhood information[12]. The learned representations capture both node attributes and local neighborhood information. Concretely, we use $\mathbf{h}_i^{(l)}$ to denote the representation of node $v_i$ in the $l$-th layer of the GNN with $\mathbf{h}_i^{(0)} = \mathbf{x}_i$. The overall representation of all the nodes in the $l$-th layer is denoted as $\mathbf{H}^{(l)}$. Let $\mathcal{N}_i = \{v_j \in \mathcal{V} \mid (v_i, v_j) \in \mathcal{E}\}$ denote the neighbor set of node $v_i$. The message-passing mechanism can be written into a unified framework, i.e., in the $l$-th layer, each node $v_i \in \mathcal{V}$ aggregates information from its neighbors as:

$$\mathbf{h}_i^{(l)} = \mathrm{UP}^{(l-1)}\Big(\mathbf{h}_i^{(l-1)}, \mathrm{AGG}^{(l-1)}\big(\{\mathbf{h}_j^{(l-1)}, \forall v_j \in \mathcal{N}_i\}\big)\Big) = \mathrm{UP}^{(l-1)}\big(\mathbf{h}_i^{(l-1)}, \mathbf{m}_{\mathcal{N}_i}^{(l-1)}\big) \quad (1)$$

where AGG is an aggregation function and UP is an update function. For example, graph convolutional network (GCN)[12] takes symmetric normalization as $\mathbf{h}_i^{(k)} = \sigma\big(\mathbf{W}^{(k)} \sum_{v_j \in \mathcal{N}_i \cup \{v_i\}} \frac{\mathbf{h}_j}{\sqrt{|\mathcal{N}_i||\mathcal{N}_j|}}\big)$, where $\mathrm{AGG}(\cdot) = \sum_{v_j \in \mathcal{N}_i} \frac{\mathbf{h}_j}{\sqrt{|\mathcal{N}_i||\mathcal{N}_j|}}$ and $\mathrm{UP}(\cdot) = \sigma\big(\mathbf{W}\frac{\mathbf{h}_i}{\mathcal{N}_i} + \mathbf{W}\mathbf{m}_{\mathcal{N}_i}\big)$.

Given the graph data $\mathcal{G} = \{\mathcal{V}, \mathcal{E}, \mathbf{X}\}$, we can utilize the aforementioned message-passing framework to get node representations. A general approach is that we adopt a GNN encoder $f$ to learn node representation as $\mathbf{H} = f(\mathbf{A}, \mathbf{X})$, which can be used for downstream tasks. For a node-level task, we aim at learning an encoder $f : \mathbf{R}^{|\mathcal{V}|\times|\mathcal{V}|} \times \mathbf{R}^{|\mathcal{V}|\times d} \to \mathbf{R}^{|\mathcal{V}|\times q}$, which computes the representation $\mathbf{H} \in \mathbf{R}^{|\mathcal{V}|\times q}$ for all the nodes. Here $d$ denotes the input feature dimension and $q$ is the node representation dimension. For graph-level task, we hope to learn a graph-level encoder $f : \mathbf{R}^{|\mathcal{V}|\times|\mathcal{V}|} \times \mathbf{R}^{|\mathcal{V}|\times d} \to \mathbf{R}^{q}$ as the representation of the whole graph. A commonly-used setting is constructing the graph-level encoder with a node-level encoder and a readout function to get graph representations[63]. An example of the readout function can be computing the sum of all node representations as the graph representation, i.e., $\mathbf{h}_{\mathcal{G}} = \sum_{v\in\mathcal{V}} \mathbf{h}_i$, where $\mathbf{h}_{\mathcal{G}}$ is the graph representation and $\mathbf{h}_i$ indicates the node representation for each node $v \in \mathcal{V}$. In this survey, we mainly focus on counterfactual learning on graphs with graph neural networks.

## 2.3 Causal inference in machine learning

Typically, machine learning algorithms heavily rely on the independent and identically distributed (i.i.d.) assumption[79] and are trained using standard empirical risk minimization. Recent works[80, 81] have shown that such machine learning algorithms can be negatively affected by spurious correlations or dependencies between observed features and class labels that only hold for certain data groups. For instance, in image classification tasks, images labeled as camels often have sand as the background, and images labeled as fish usually have water as the background. Consequently, the empirical risk minimization classifier may rely on the spurious background pattern for predictions rather than the objects in the images. However, this spurious correlation does not indicate causality[41] and may severely limit the robustness, explainability, and generalization ability of machine learning models[82]. For example, such a model may misclassify cows on sand as camels or birds flying over water as fish during testing. In real-world scenarios, inaccurate predictions caused by spurious correlations can bring risks in high-stake environments such as healthcare. For example, it might lead to wrong medical interventions, posing threats to human life. Hence, there is a growing trend to incorporate causal inference in machine learning

methods to learn causal features and improve generalization across different testing distributions. For instance, in an animal classification task, a classifier that focuses on the camel pattern rather than the sand pattern for classification can perform well for camels with various backgrounds[83].

Compared with traditional statistical models, causal models have better generalization ability in modeling real-world systems[82]. According to Pearl's causal hierarchy[84] and the description about levels of causal modeling in [85], there are four levels of models to represent the real-world systems as shown in Table 3. The physical model of a set of coupled differential equations is a quite comprehensive description of a system. Differential equations allow us to predict the future behavior of a physical system and give us physical insights into the functioning[85]. Compared with the physical models, a statistical model is quite superficial since it can only model the correlation between variables and when the input distribution changes the model performance may degrade dramatically[86]. Causal models are between these two levels of models. When we have a graphical model with all the edges as causal edges endowed with the notion of direct causal effect, the model would be a causal graphical model and allow us to perform interventions on variables. Learning from interventions can help models learn invariant mechanisms against distribution shifts[87]. Go beyond the causal graphical model, a structural causal model (SCM) is composed of a set of causal variables and a set of structural equations with noise distributions[88]. With SCMs, we can not only get the interventional distribution but also perform counterfactual reasoning[84]. As expressed in [84], we can replace the parents-child relationship $p(X_i \mid \mathbf{PA}_i)$ with its functional counterpart $X_i = f_i(\mathbf{PA}_i, U_i)$ in Bayesian networks to get the structural causal models, where $\mathbf{PA}_i$ denotes the parent variables of $X_i$. The definition of the structural causal model is given as:

**Definition 5** (structural causal model (SCM)[87]). To describe the causal relationship between a set of $n$ random variables $\mathcal{X} = \{X_1, \cdots, X_n\}$, an SCM $\mathcal{M} = (\mathcal{F}, p_{\mathcal{U}})$ consists of 1) a set $\mathcal{F}$ of $n$ assignments (the structural equations):

$$\mathcal{F} = \{X_i := f_i(\mathbf{PA}_i, U_i)\}_{i=1}^{n} \tag{2}$$

where $f_i$ is a deterministic function which is used for computing each variable $X_i$ from its causal parents $\mathbf{PA}_i \subseteq \mathcal{X} \backslash \{X_i\}$ and an exogenous noise variable $U_i$. The assignment symbol ":=" stands for defining, which is used to indicate the asymmetry of the causal relationship; and 2) a joint distribution $p_{\mathcal{U}}(U_1, \cdots, U_n)$ over the exogenous noise variables.

Causal inference provides the theoretical foundation for identifying and estimating causal effects. Counterfactual learning is built on causal inference.

Table 3 A simple taxonomy on causal models

| Models | i.i.d. | Intervention | Counterfactual |
|---|---|---|---|
| Physical | Yes | Yes | Yes |
| Structural causal | Yes | Yes | Yes |
| Causal graphical | Yes | Yes | No |
| Statistical | Yes | No | No |

The definition of interventional distribution in the SCM framework is quite natural. To model an intervention, we can replace the corresponding structural equation and consider the resulting entailed distribution[85]. Counterfactuals are quite different from observations and interventions. Observations are used to describe what is passively seen or measured, and interventions are used to describe active external manipulation or experimentation, while counterfactuals indicate what would or could have been, given that something else was, in fact, observed[87]. Counterfactuals help us better understand causal relationships in graph structures.

The formal definition of counterfactuals is given as:

**Definition 6** (counterfactuals in SCMs[87]). Given evidence $\mathbf{X} = \mathbf{x}$ observed from an SCM $\mathcal{M} = (\bar{\mathcal{F}}, p_{\mathcal{U}})$, the counterfactual SCM $\mathcal{M}^{\mathbf{X}=\mathbf{x}}$ is obtained by updating $p_{\mathcal{U}}$ with its posterior: $\mathcal{M}^{\mathbf{X}=\mathbf{x}} = (\mathcal{F}, p_{\mathcal{U}|\mathbf{X}=\mathbf{x}})$. Counterfactuals are then computed by performing interventions in the counterfactual SCM $\mathcal{M}^{\mathbf{X}=\mathbf{x}}$. The intervention $do(X_i := x_i)$ is modeled by replacing the $i$-th structural equation in $\mathcal{F}$ by $X_i := x_i$, yielding the intervened SCM $\mathcal{M}^{do(X_i:=x_i),\mathbf{X}=\mathbf{x}} = (\mathcal{F}', p_{\mathcal{U}|\mathbf{X}=\mathbf{x}})$. The interventional distribution $p(\mathbf{X}_{-i} \mid do(X_i = x_i))$, where $\mathbf{X}_{-i} = \mathbf{X} \backslash \{X_i\}$, and intervention graph $\mathcal{G}'$ are those induced by $\mathcal{M}^{do(X_i:=x_i),\mathbf{X}=\mathbf{x}}$.

Here, $do(X_i := x_i)$ is to emphasize the difference between passive observation and active intervention, denoting an intervention by which variable $X_i$ is set to value $x_i$[84].

Counterfactual learning is a learning scheme with the aid of counterfactuals. Causal inference community focuses on estimating the counterfactuals[84]. Machine learning community leverages counterfactual estimation as one module of the model[47, 89], and designs different modules besides the counterfactual estimation module to achieve the goal of fairness[89], interpretation[47] or link prediction and recommender systems[30].

## 3 Counterfactual fairness on graphs

In many real-world applications, training data may contain demographic biases due to societal or historical factors. Machine learning models trained on such data can inherit these biases and produce unfair predictions, as

seen in credit card decision-making tasks where models can exhibit gender or race biases[44, 90]. Biased predictions can result in systemic discrimination and undermine public trust in machine learning models, which has led to growing interest in fairness-aware machine learning[44]. Dai et al.[33] highlight that fairness issues are more severe in graphs, where both node features and graph structure can contribute to bias. As many real-world systems rely on graph neural networks, unfair predictions can have serious consequences. Researchers have proposed methods to ensure fairness in graph learning[28, 89, 91–93]. However, existing fairness notions are predominantly correlation-based and may not detect statistical anomalies such as Simpson's paradox[48]. To address this issue, counterfactual learning has emerged as a promising approach for building fair machine learning models that achieve counterfactual fairness[34, 48]. Counterfactual fairness is based on the idea that a prediction for an individual is fair if it remains the same in a counterfactual world where the individual belongs to a different demographic group[34]. Counterfactual fairness on graphs is attracting increasing attention[32, 89, 92, 94]. In this section, we introduce the background of fairness issues in graph learning and counterfactual fairness, followed by a general framework of graph counterfactual fairness models and their details. We conclude with widely used fairness evaluation metrics and datasets.

## 3.1 Background of graph counterfactual fairness

### 3.1.1 Sources of biases

Various biases are prevalent in real-world data sources, and machine learning models trained on biased datasets can easily produce unfair predictions. Bias in machine learning can occur at different stages, such as data, algorithm and user interaction[44, 95]. We can categorize commonly observed bias in i.i.d. data into four categories: historical bias, representation bias, temporal bias and attribute bias[33]. Specifically, historical bias refers to a bias in data due to historical reasons, such as gender or race bias. Representation bias denotes the discrepancy between collected data points and real-world data points, which arises from the under-representation of the collected data. Temporal bias results from changes over time. These biases are widely present in image datasets[96], text datasets[97], and video datasets[98]. Similar to i.i.d data, graphs also exhibit the aforementioned biases in their node attributes.

In addition to the biases in node attributes, graph-structured data also has distinctive biases due to topology structures such as the linking bias and the structural bias[33]. Linking bias happens since nodes have intrinsic preferences to connect with others, e.g., nodes of similar ages are more likely to connect to each other. Structural bias is caused due to the information propagation between nodes on graph-structure data. The information propagation can amplify the sensitive information on each node[44]. In real-world networks, nodes with the same sensitive attribute (e.g., ages) are more likely to get connected. For instance, young people tend to make friends with people of similar ages on social networks. The message-passing of GNNs will further magnify this bias due to the smoothing nature of GNN, i.e., representations of nodes of the same sensitive attribute will be more similar while representations of nodes of different sensitive attributes will be more different after the aggregation. It will make the predictions for downstream tasks highly correlated with sensitive attributes because the learned node representations are highly dependent on the sensitive attributes of nodes[33]. GNNs are commonly used on graph-structure data and the message-passing framework of GNNs can mix the biases stored in node attributes and graph structures together into the node representations[91], which makes the bias issue more severe.

### 3.1.2 Definition of graph counterfactual fairness

In machine learning area, biases are measured from different perspectives. Group fairness and individual fairness are two commonly used algorithmic bias metrics[31]. Group fairness aims to ensure that any demographic group should not get discriminatory treatment. Here a group means a set of people that have the same sensitive attribute, e.g., people of the same gender. Instead of focusing on demographic statistics, individual fairness aims to ensure that two similar individuals should receive similar algorithmic treatment[33]. For example, two people who have similar credit records should get a similar credit loan.

Kusner et al.[34] find that group fairness and individual fairness have some issues of incompatibility and they only focus on subpopulations or individuals in the actual world. These fairness notions rely on the statistical correlation among variables, which may not describe the intrinsic causal structure[99]. They also fail to capture the discrepancy when an individual belonging to the actual world and a counterfactual world. For instance, in real-world, people tend to come up with counterfactual questions, like "what if I am a girl, will I be admitted to xxx university?". A system that cannot answer counterfactual questions and fulfill counterfactual fairness is not reliable and can severely damage users' experience[48]. Inspired by the development of counterfactual learning, counterfactual fairness is proposed as a notion to measure fairness from the perspective of causal inference[34]. Counterfactual fairness makes the outputs of a machine learning model for an individual in the actual world remain the same when we flip the sensitive attribute of the same individual to the counterfactual world. We give an example to illustrate the notion of counterfactual fairness. Many companies use machine learning and graph neural

networks to assign credit loans to applicants. However, the model may give unfair predictions due to the bias in the training data. For example, a model may predict that a white person should get a large loan based on his/her features; While when we flip the person's ethnicity to black, the model gives the decision that the person can only get a small loan, which is counterfactually unfair. In this example, we do not compare the individual's outcome with other groups of people or any other individuals but just the same individual in a counterfactual world. Following [89], we extend the formal definition of graph counterfactual fairness as:

**Definition 7** (graph counterfactual fairness[89]). An encoder $\Phi(\cdot)$ or classifier $\phi(\cdot)$ satisfies graph counterfactual fairness if for any node $v_i$:

$$\begin{aligned} &P\left((\mathbf{z}_i)_{S\leftarrow s} \mid \mathbf{X}, \mathbf{A}\right) = P\left((\mathbf{z}_i)_{S\leftarrow s'} \mid \mathbf{X}, \mathbf{A}\right) \quad \text{or} \\ &P\left((\hat{y}_i)_{S\leftarrow s} \mid \mathbf{X}, \mathbf{A}\right) = P\left((\hat{y}_i)_{S\leftarrow s'} \mid \mathbf{X}, \mathbf{A}\right) \quad \forall s \neq s' \end{aligned} \tag{3}$$

where $s, s' \in \{0,1\}^n$ are two arbitrary sensitive attribute values of all nodes. $\mathbf{z}_i = (\Phi(\mathbf{X}, \mathbf{A}))_i$ denotes the node representations for node $v_i$ and $\hat{y}_i = \phi(\mathbf{z}_i)$ denotes the predicted label for node $v_i$. In other words, given a graph $\mathcal{G} = \{\mathbf{A}, \mathbf{X}\}, \Phi(\cdot)$ should be optimized by minimizing the distribution discrepancy between the representations $(\Phi(\mathbf{X}_{S\leftarrow s}, \mathbf{A}_{S\leftarrow s}))_i$ and $(\Phi(\mathbf{X}_{S\leftarrow s'}, \mathbf{A}_{S\leftarrow s'}))_i$. $\phi(\cdot)$ should be optimized by minimizing the distribution discrepancy between the predictions $(\phi(\mathbf{X}_{S\leftarrow s}, \mathbf{A}_{S\leftarrow s}))_i$ and $(\phi(\mathbf{X}_{S\leftarrow s'}, \mathbf{A}_{S\leftarrow s'}))_i$.

Here we use $\mathbf{X}_{S\leftarrow s'}$ to denote a specific value of the counterfactual, i.e., "what would the node feature have been if the sensitive attribute of the nodes had been set as $s'$, given the original node features $\mathbf{X}$ and graph structure $\mathbf{A}$." $\mathbf{A}_{S\leftarrow s'}$ is also got in a similar manner[89].

## 3.2 Methods of graph counterfactual fairness

To eliminate the bias in graph-structure data, extensive efforts have been put on building fairness-aware GNNs[33]. Generally, existing debiasing methods can be roughly categorized into three categories, i.e., adversarial debiasing methods[100, 101], fairness constrain methods[28, 102, 103], counterfactual-based methods[32] and other methods, such as random walk-based method[104, 105] and so on. Specifically, adversarial debiasing methods[100, 101] utilize adversarial learning to learn node representations that are independent of the sensitive attribute. Fairness constrain methods[28, 102, 103] directly add carefully designed fairness constraints such as the covariance between sensitive attributes and predicted labels to the objective function to achieve fairness. For a comprehensive overview of these debiasing techniques, please refer to [33]. In this survey, we focus on counterfactual-based methods, which leverage counterfactual reasoning to ensure GNN fairness. Counterfactual fairness on graphs overcomes the inherent limitations of correlation-based fairness notions such as group fairness. However, counterfactual fairness introduces new challenges, including constructing causal relationships among variables to obtain counterfactuals and achieving counterfactual fairness with observed data and estimated counterfactuals. Additionally, counterfactual fairness in graph-structured data presents unique technical challenges. The nodes in graphs are interconnected, leading to a non-independent and non-identically distributed nature in data generation. In this section, to provide insight into counterfactual learning on graphs, we first introduce a general framework for achieving counterfactual fairness on graphs. Following that, we will discuss representative and state-of-the-art works in detail.

### 3.2.1 General framework of counterfactual fairness

Inspired by the advances of counterfactual learning, several methods have been proposed to achieve counterfactual fairness on graph-structure data[31, 32, 89, 92, 94, 106]. In general, most of these methods can be unified into a two-step framework designed to address the aforementioned challenges. Fig. 3 illustrates the general framework, using node $v_i$ as an example. First, counterfactual aug-

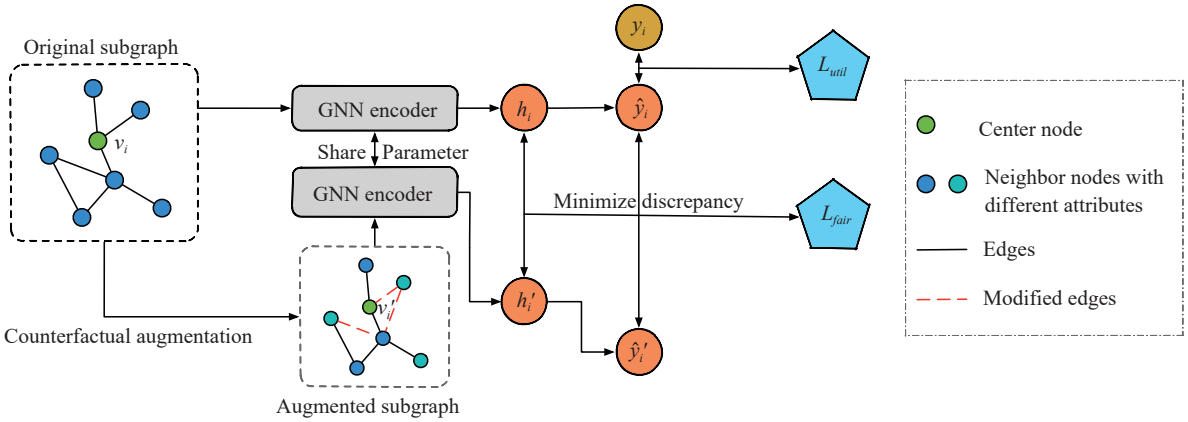



Fig. 3 Overall framework of graph counterfactual fairness (Colored figures are available in the online version at https://link.springer.com/journal/11633)

mentations are generated by flipping the sensitive attributes and related information of observed data. Second, both observed factual and generated counterfactual samples are fed into graph neural networks to obtain factual and counterfactual representations while minimizing the discrepancy between them to achieve counterfactual fairness.

**Counterfactual augmentation.** In graph counterfactual fairness, we want to mitigate the bias caused by the sensitive information, which means the counterfactuals should have the opposite sensitive information. Thus, a straightforward way is to flip the sensitive attributes of observed samples to get the counterfactuals. However, this method only works when we have the assumption that the sensitive attributes do not have any causal effect on the generation of other attributes and graph structures, which is not valid in most situations. To solve this issue, various augmentation methods[92, 107–109] to model the dependency between graph structures and sensitive attributes are proposed. Here, we summarize these augmentation methods into a unified framework.

Let $\mathbf{s} = \{s_i\}_{i=1}^n$ represent the vector of sensitive attributes, where $s_i \in \{0, 1\}$ is the sensitive attribute of node $v_i$. Let $\mathbf{X}_{S \leftarrow s'}$ denote the counterfactual features when the sensitive attributes are flipped to $s'$. This corresponds to "what would the node features have been if the sensitive attributes of the nodes had been set as $s'$, given the original node features $\mathbf{X}$ and graph structure $\mathbf{A}$?". Thus, we can represent the actual counterfactual graph as $(\mathbf{A}_{\mathbf{S} \leftarrow s'}, \mathbf{X}_{\mathbf{S} \leftarrow s'})$, where $s'$ represents the flipped sensitive attributes of $s$. Existing counterfactual fairness methods use augmentations to approximate the counterfactual graph. Considering the counterfactual augmentation as $\mathcal{T}(\cdot)$, which can modify the graph structure or feature matrix, we define the approximated counterfactual graph as $\mathcal{G}' = (\mathbf{A}', \mathbf{X}')$. For simplicity, in the following sections, we use the term "counterfactual graph" to denote the approximated counterfactual graph. The counterfactual graph is obtained by:

$$\mathbf{A}', \mathbf{X}' = \mathcal{T}(\mathbf{A}, \mathbf{X}) \tag{4}$$

where $\mathcal{T}(\cdot)$ denotes the counterfactual augmentation that flips sensitive attributes and sensitive-related information, e.g., $\mathcal{T}(\cdot)$ can be just flipping the sensitive attribute of a node or flipping the sensitive attributes of neighbor nodes in each ego-graph.

**Counterfactual regularization.** For graph counterfactual fairness, we want to minimize the discrepancy of the node representations for factual graph $\mathcal{G}$ and counterfactual graph $\mathcal{G}'$, which makes the representation independent of sensitive attributes. Many works[89, 92] adopt the Siamese neural networks to learn fair node representations as:

$$\mathbf{H} = f_\theta(\mathbf{A}, \mathbf{X}), \quad \mathbf{H}' = f_\theta(\mathbf{A}', \mathbf{X}') \tag{5}$$

where $f_\theta$ denotes the graph neural network with parameter $\theta$. $\mathbf{H}$ is the factual node representation and $\mathbf{H}'$ stands for the counterfactual node representation. To minimize the discrepancy between factual and counterfactual node representations, the objective function can be written as:

$$\mathcal{L}_{\text{fairness}} = \text{dist}(\mathbf{H}, \mathbf{H}') \tag{6}$$

where $\text{dist}(\cdot, \cdot)$ is the distance measure such as the negative cosine similarity and L2 distance[89, 92].

**Overall objective.** For graph counterfactual fairness models, the overall objective function is:

$$\min_\theta \mathcal{L}_{\text{utility}} + \beta \mathcal{L}_{\text{fairness}} \tag{7}$$

where $\beta$ is the hyper-parameter to control the trade-off between utility and fairness. $\mathcal{L}_{\text{utility}}$ is the loss function for the utility of the model such as node classification loss and graph classification loss. $\mathcal{L}_{\text{fairness}}$ is the graph counterfactual fairness constraining $\mathcal{L}_{\text{fairness}} = \text{dist}(f_\theta(\mathbf{A}, \mathbf{X}), f_\theta(\mathcal{T}(\mathbf{A}, \mathbf{X})))$. $\theta$ is the set of model parameters to be learned.

### 3.2.2 Methodologies

The aforementioned general framework can be easily extended to different scenarios. For instance, in the augmentation step, different kinds of augmentation techniques can be applied, such as random perturbation[92], parameterized augmentation[89], heuristic-based augmentation[92, 107, 108] and data-based[106]. For the counterfactual regularization, various distance measures on distributions can be used, such as triplet loss function[89]. The utility function can be the classification or regression losses, depending on downstream tasks[92, 110]. Next, we will introduce details of representative graph counterfactual fairness methods. Table 4 summarizes existing graph counterfactual fairness papers.

**NIFTY[92].** To build trustworthy models on graph data, NIFTY[92] tries to establish a key connection between fairness and stability and learn fair and stable node representations at the same time. It defines three kinds of perturbations on node attributes, sensitive attributes and graph structure. In NIFTY, the counterfactual augmentation is obtained as:

$$\mathbf{A}', \mathbf{X}' = \mathcal{T}(\mathbf{A}, \mathbf{X}) = \mathbf{A}, \mathbf{X}_{\text{flip}(\mathbf{s})} \tag{8}$$

where $\mathcal{T}(\cdot)$ denotes the counterfactual augmentation. NIFTY just flips the sensitive attributes $\mathbf{s}$ in the feature matrix $\mathbf{X}$. Then, together with the randomized perturbation on node attributes and graph structure, different views are fed into GNN-based encoder to get node representations. Then, NIFTY adopts a triplet-based objective to maximize the similarity between the original representation and its augmented (i.e., counterfactual and noisy) representations to ensure that

Table 4 Summary of graph counterfactual fairness papers

| Methods | Edge aug | Feature aug | Learnable mask | Stage | Subjects | Causal graph |
|---|---|---|---|---|---|---|
| GEAR[89] | ✓ | ✓ | ✓ | In-processing | Subgraph | Yes |
| MCCNIFTY[94] | ✓ | ✓ | ✓ | In-processing | Subgraph | No |
| NIFTY[92] | ✓ | ✓ | × | In-processing | Whole graph | No |
| Fairness-aware[32] | ✓ | ✓ | ✓ | In-processing | Whole graph | No |
| CAF[106] | × | × | × | In-processing | Subgraph | Yes |
| RFCGNN[111] | × | × | × | In-processing | Subgraph | Yes |
| RFCGNN+[112] | × | × | × | In-processing | Subgraph | Yes |

the representations are independent of sensitive attributes. The loss function can be written as:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{utility}} + \mathcal{L}_{\text{fairness}} = \frac{1}{|\mathcal{V}_L|}\sum_{v_i\in\mathcal{V}_L} l(f_\Phi(\mathbf{h}_i), y_i) + \frac{1}{2|\mathcal{V}|}\sum_{v_i\in\mathcal{V}} \big(\text{dist}(t_\phi(\mathbf{h}_i), \text{sg}(\mathbf{h}'_i)) + \text{dist}(t_\phi(\mathbf{h}'_i), \text{sg}(\mathbf{h}_i))\big)$$

where $f_\Phi(\cdot)$ is the prediction head, i.e., $\hat{y}_i = f_\Phi(\mathbf{h}_i)$. $t_\phi(\cdot)$ denotes an additional projection head. $\text{sg}(\cdot)$ stands for the stop-gradient operation. Here $\text{dist}(\cdot,\cdot)$ is cosine distance.

**GEAR[89].** Many existing works neglect the causal interaction between a node and its neighbors[92, 113]. Thus, Ma et al.[89] propose a framework GEAR to generate counterfactual data with the causal interaction between nodes and their neighborhood into consideration. Concretely, they utilize self-perturbation and neighbor-perturbation to flip the sensitive information in each node's ego-graph to generate counterfactuals. Self-perturbation just flips the sensitive attribute of the center node. Neighbor-perturbation modifies the sensitive attribute of neighbor nodes. Aligned with our unified framework, the augmentation process is modeled as:

$$\mathbf{A}', \mathbf{X}' = \mathcal{T}_{\text{VGAE}}(\mathcal{T}_{\text{perturb}}(\mathbf{A}, \mathbf{X})) \tag{9}$$

where $\mathcal{T}_{\text{perturb}}$ represents self-perturbation $\mathcal{T}_{\text{self}}$ and neighbor-perturbation $\mathcal{T}_{\text{neighbor}}$. These perturbations can help to find counterfactuals which have a different sensitive attribute in itself or neighbors. $\mathcal{T}_{\text{VGAE}}$ is a variational graph auto-encoder which helps to model the dependencies in the latent representation level so that the model can flip the causally-related information in ego-graphs. Concretely, the two kinds of counterfactuals can be generated by:

$$\begin{aligned}
&\mathbf{X}_{S_i=s'} = \mathcal{T}_{\text{self}}(\mathbf{X})\\
&(\mathbf{A}', \mathbf{X}') = \mathcal{T}_{\text{VGAE}}(\mathbf{A}, \mathbf{X}_{S_i=s'})\\
&\mathbf{X}_{\mathcal{N}_i=s'} = \mathcal{T}_{\text{neighbor}}(\mathbf{X})\\
&(\mathbf{A}'', \mathbf{X}'') = \mathcal{T}_{\text{VGAE}}(\mathbf{A}, \mathbf{X}_{\mathcal{N}_i=s'}).
\end{aligned} \tag{10}$$

With the sensitive-information-flipped counterfactuals, GEAR minimizes the discrepancy between factual representation and counterfactual representation so that the model can find the shared information for observed data and counterfactuals. The fairness regularization is written as:

$$\mathcal{L}_{\text{fairness}} = \frac{1}{N}\sum_{i=1}^{N}\Big((1-\lambda_s)\, d\left(\mathbf{h}_i, \mathbf{h}'_i\right) + \lambda_s d\left(\mathbf{h}_i, \mathbf{h}''_i\right)\Big) \tag{11}$$

where $\mathbf{H} = f_\theta(\mathbf{A}, \mathbf{X}), \mathbf{H}' = f_\theta(\mathbf{A}', \mathbf{X}')$ and $\mathbf{H}'' = f_\theta(\mathbf{A}'', \mathbf{X}'')$. The obtained representation can be used for various kinds of downstream tasks. For node classification, the utility loss function is:

$$\mathcal{L}_{\text{utility}} = \frac{1}{|\mathcal{V}_L|}\sum_{v_i\in\mathcal{V}_L} l(f_\Theta(\mathbf{h}_i), y_i) \tag{12}$$

where $\mathcal{V}_L \in \mathcal{V}$ is the set of labeled nodes. $f_\Theta(\cdot)$ is parameterized with $\theta'$ and make predictions for node classification task, i.e., $\hat{y}_i = f_\Theta(\mathbf{h}_i)$. $l(\cdot,\cdot)$ is the cross entropy loss function.

**MCCNIFTY[94].** Building upon NIFTY[92], MCCNIFTY[94] employs the same perturbation techniques but with a focus on subgraph-level processing to improve computational efficiency. Specifically, MCCNIFTY first utilizes the personalized PageRank algorithm (PPP) to extract informative subgraphs for each node, and then applies perturbations to these subgraphs. Using these subgraphs and their counterfactual counterparts, MCCNIFTY designs a multi-view uncertainty-aware node embedding learning module inspired by evidential theory. This approach has the advantage of estimating uncertainty scores that help control the information propagation process of GNN, allowing the model to adaptively adjust edge attention and defend against adversarial attacks[92]. With the uncertainty-aware GNN encoder, MCCNIFTY follows NIFTY's pipeline and designs a loss function to maximize the similarity of multi-view node embeddings. The primary distinctions between MCCNIFTY and NIFTY are: 1) MCCNIFTY focuses on subgraph-level processing for enhanced efficiency, and 2)

MCCNIFTY adopts an uncertainty quantification module to help defend against adversarial attacks.

**Fairness-aware[32].** As most existing counterfactual augmentation approaches simply adopt the randomized perturbation on feature matrix and adjacency matrix, Köse and Shen[32] propose to leverage the knowledge from the data to adaptively augment the features and graph structures. As GNNs can propagate and amplify biases in graph-structured data, adaptive augmentation under suitable supervision can help to reduce possible bias inherited in a graph structure and node features. Concretely, the adaptive counterfactual augmentation can be modeled as:

$$\mathbf{A}', \mathbf{X}' = \mathcal{T}(\mathbf{A}, \mathbf{X}) = \text{DropEdge}(\mathbf{A}), \text{MaskFeat}(\mathbf{X}) \tag{13}$$

where $\text{MaskFeat}(\mathbf{X}) = \{\mathbf{m} \circ \mathbf{x}_1, \cdots, \mathbf{m} \circ \mathbf{x}_N\}$ and $\mathbf{m} \in \{0,1\}^F$ is the mask vector. Each entry $m_i$ of $\mathbf{m}$ is drawn independently from a Bernoulli distribution determined by the correlation score between the nodal features and the sensitive attributes of the node. The intuition is that features with a high correlation with the sensitive attribute will tend to induce an unfair prediction. $\text{DropEdge}(\mathbf{A})$ assigns different edge-dropping probabilities for edges to get different views. For an edge $e_{ij}$, the adaptive edge deletion probabilities are designed as:

$$p\left(e_{ij}^{(1)}\right) = \begin{cases} p_1, & \text{if } s_i = s_j \\ p_2, & \text{if } s_i \neq s_j \end{cases}$$
$$p\left(e_{ij}^{(2)}\right) = \begin{cases} p_3, & \text{if } s_i \neq s_j \\ p_4, & \text{if } s_i = s_j \end{cases} \tag{14}$$

where the superscript (1) and (2) are used to denote augmented view 1 and view 2, respectively. $p(e_{ij}^{(1)})$ and $p(e_{ij}^{(2)})$ are the deletion probabilities for the edge $e_{ij}$ in two views. The probabilities should follow $p_1 > p_2$ and $p_3 < p_4$ so that the edges connecting two nodes of the same sensitive attribute have a larger probability to be deleted than edges that connect two nodes of different sensitive attributes, which can alleviate the structural bias of the graph. With two different views of less bias, it performs contrastive learning to learn node representation as:

$$\ell\left(\mathbf{h}_i^1, \mathbf{h}_i^2\right) = -\log \frac{\mathrm{e}^{s\left(\mathbf{h}_i^1, \mathbf{h}_i^2\right)/\tau}}{Z} \tag{15}$$

where

$$Z = \sum_{k=1}^{N} \mathbb{1}_{[k \neq i]} \mathrm{e}^{s\left(\mathbf{h}_i^1, \mathbf{h}_k^2\right)/\tau} + \mathrm{e}^{s\left(\mathbf{h}_i^1, \mathbf{h}_i^2\right)/\tau} + \sum_{k=1}^{N} \mathbb{1}_{[k \neq i]} \mathrm{e}^{s\left(\mathbf{h}_i^1, \mathbf{h}_k^1\right)/\tau}$$

$s(\cdot,\cdot)$ is a score function, i.e., cosine similarity. $\tau$ is the temperature parameter and $\mathbb{1}_{[k \neq i]} \in \{0,1\}$ is the indicator function and takes the value 1 if and only if $k \neq i$. The final fairness regularization can be written as $\mathcal{L}_{\text{fairness}} = \frac{1}{2N} \sum_{i=1}^{N} \left[\ell\left(\mathbf{h}_i^1, \mathbf{h}_i^2\right) + \ell\left(\mathbf{h}_i^2, \mathbf{h}_i^1\right)\right]$ to be in a symmetric form, where $\ell(\cdot,\cdot)$ is the contrastive loss from (15).

**CAF[106].** CAF claims that random perturbation or generative approach have their own drawbacks. It proposes to obtain counterfactuals within the training data using sensitive attributes and labels as guidance rather than random perturbation or generation. Specifically, CAF finds counterfactual graphs for a given node by searching in the latent space rather than directly searching graphs, which is complex and infeasible. To achieve this, CAF first uses one GNN to learn latent content representation $\mathbf{C}$ and environment representation $\mathbf{E}$ as:

$$[\mathbf{C}, \mathbf{E}] = \mathbf{H} = f_\theta(\mathbf{A}, \mathbf{X}). \tag{16}$$

To make content representation $\mathbf{C}$ have enough information for the downstream tasks, they introduce a classifier $f_\theta$ to predict labels from $\mathbf{C}$. The utility loss function is:

$$\mathcal{L}_{\text{utility}} = \frac{1}{|\mathcal{V}_L|} \sum_{v_i \in \mathcal{V}_L} l(f_\Theta(\mathbf{c}_i), y_i) \tag{17}$$

where $\mathcal{V}_L$ is a set of labeled nodes and $l(f_\Theta(\mathbf{c}_i), y_i)$ means the cross entropy loss.

To ensure $\mathbf{C}$ contains enough information to reconstruct the observed graph, CAF uses a sufficiency constraint:

$$\mathcal{L}_{\text{suf}} = \frac{1}{|\mathcal{E}| + |\mathcal{E}^-|} \sum_{(v_i, v_j) \in \mathcal{E} \cup \mathcal{E}^-} -e_{ij} \log p_{ij} - (1 - e_{ij}) \log p_{ij} \tag{18}$$

where $\mathcal{E}^-$ is sampled negative edges and $p_{ij}$ is the link existence probability between node $i$ and node $j$. CAF also uses orthogonal constraint to ensure that $\mathbf{c}_i$ and $\mathbf{e}_i$ are disentangled, i.e. $\mathbf{c}_i^{\mathrm{T}} \mathbf{e}_i = 0$.

With the disentangled node representation, for node $v_i$, its counterfactual with sensitive attribute $s_i$ unchanged while $y_i$ changed can be found as:

$$\mathbf{h}_i^c = \arg\min_{\mathbf{h}_j \in \mathcal{H}} \left\{ \|\mathbf{h}_i - \mathbf{h}_j\|_2^2 \mid y_i \neq y_j, s_i = s_j \right\} \tag{19}$$

where $\mathcal{H} = \{\mathbf{h}_i \in \mathcal{V}\}$. Similarly, its counterfactual with $y_i$ unchanged but $s_i$ changed can be found as:

$$\mathbf{h}_i^e = \arg\min_{\mathbf{h}_j \in \mathcal{H}} \left\{ \|\mathbf{h}_i - \mathbf{h}_j\|_2^2 \mid y_i = y_j, s_i \neq s_j \right\} \tag{20}$$

To make $\mathbf{e}_i$ be invariant to the content information in

$\mathbf{c}_i$, CAF defines an invariance constraint:

$$\mathcal{L}_{\text{inv}} = \frac{1}{|\mathcal{V}| \times K} \sum_{v_i \in \mathcal{V}} \sum_{k=1}^{K} [dis(\mathbf{c}_i, \mathbf{c}_i^{e_k}) + dis(\mathbf{e}_i, \mathbf{e}_i^{c_k}) + \gamma K \times |\cos(\mathbf{c}_i, \mathbf{e}_i)|] \tag{21}$$

where $\mathbf{c}_i^{e_k}$ stands for counterfactual content representation and $\mathbf{e}_i^{c_k}$ stands for the counterfactual environment representation. $K$ means the number of counterfactuals for a node.

The fairness regularization can be written as:

$$\mathcal{L}_{\text{fairness}} = \alpha \mathcal{L}_{\text{suf}} + \beta \mathcal{L}_{\text{inv}}. \tag{22}$$

Very similar to CAF[106], RFCGNN[111] aims to alleviate biases caused by sensitive attributes and the labeling process. Similar to CAF, RFCGNN divides node features into two parts: sensitive relational representation $\mathbf{H}_S$ and content representation $\mathbf{H}_C$. The major components in RFCGNN are almost the same as CAF, such as the component of finding counterfactuals and the orthogonal constraint. Compared with CAF, RFCGNN introduces two extra components.

The first one is a constraint to remove the biases in $\mathbf{H}_S$. It first maps it to a new space using the following equation[111]:

$$P(W = w_i | \mathbf{H}_S) = \frac{\exp(-dis(\mathbf{H}_S, p_{w_i}))}{\sum_{j=1}^{W} \exp(-dis(\mathbf{H}_S, p_j))} \tag{23}$$

where $W$ is a multinomial random variable and $dis(,)$ means Euclidean distance. We use $P_i$ to represent $P(W = w_i|\mathbf{H}_S)$ for simplicity. This constraint in RFCGNN can be defined as:

$$\mathcal{L}_F = \left( \sum_{u=1}^{W} \left| \frac{1}{|S^+|} \sum_{v_i \in S^+} P_i - \frac{1}{|S^-|} \sum_{v_i \in S^-} P_i \right| + \sum_{i=1}^{n} (\mathbf{H}_{S_i} - \overline{\mathbf{H}}_{S_i})^2 \right) \tag{24}$$

where $S^+$ means favored groups, $S^-$ means deprived groups and $\overline{\mathbf{H}}_{S_i}$ stands for the reconstruction of $\mathbf{H}_{S_i}$ using $W$. It also introduces a self-paced learning module to address the problem of label scarcity.

RFCGNN+[112] is an extension of RFCGNN. Compared with RFCGNN, RFCGNN+ designs an extra component named fair subgraph encoder module to alleviate the bias caused by the aggregation process in GNNs.

### 3.3 Taxonomy of graph counterfactual fairness methods

In this subsection, we provide a detailed categorization of graph counterfactual fairness models. First, we categorize models based on the stage at which fairness is addressed. Then, we classify existing works according to research subjects and whether they model causal graphs or not. Table 4 summarizes this categorization.

**1) Stages.** Based on the stage at which fairness is addressed, existing graph fairness models can be classified into pre-processing, in-processing, and post-processing methods[33]. It should be noted that, currently, no works fall into the pre-processing and post-processing methods categories. Most existing works are in-processing methods[32, 89, 92, 94, 106, 111]. These methods utilize carefully designed GNNs to obtain fair node representations, which help minimize the discrepancy between factual and counterfactual node representations. Researchers have developed various loss functions to improve performance on different downstream tasks.

**2) Subjects.** Based on the augmentation subject, existing works can be categorized into subgraph-based methods and entire graph-based methods. Subgraph-based methods[89, 94, 106, 111, 112] generate counterfactuals using target nodes′ subgraphs. GEAR [89] samples ego-centric subgraphs for individual nodes, considering only causal relationships within subgraphs. It employs a personalized PageRank algorithm to compute node importance scores, selecting the most influential neighbors for each central node. MCCNIFTY[94] similarly uses personalized PageRank scores to extract informative regional substructures for each node, but its aim is to make node operations more computationally friendly and reduce noise from the global structure. CAF[106], RFCGNN[111] and RFCGNN+[112] also sample ego subgraphs for each node. Whole graph-based methods[31, 32, 92] treat the entire graph as a whole and flip all nodes′ sensitive attributes. This approach reduces computation costs but fails to model the intrinsic causal structures of the observational data.

**3) Causal graphs.** We categorize existing graph counterfactual fairness works into causal-graph-aware methods and causal-graph-unaware methods. Most of the existing works[31, 32, 92, 94] just perturb the sensitive attributes to get the potential outcomes. These works don′t give an explicit causal graph or structural equations. Hence, we categorize them as causal-unaware methods. In contrast, GEAR[89] constructs an explicit causal graph to model the intrinsic causal dependency of the variables in the ego-centric subgraphs for each node. Then they use variational graph auto-encoder to get rid of the causal effects of the sensitive attributes. CAF[106] builds a structural causal model to present the causal relationships among the following five variables: sensitive attribute, ground-truth label, environment feature, content feature, and ego-graph for each node. RFCGNN[111] also uses a structural model which consists of five kinds of elements.

### 3.4 Evaluation metrics for graph counterfactual fairness

To evaluate models for graph counterfactual fairness, we consider both model utility and bias mitigation. Model utility evaluation metrics vary depending on the downstream tasks, with accuracy[89], AUC[92], and F1 score[92] commonly used for node classification tasks. Models achieving graph counterfactual fairness should also satisfy other fairness constraints, so group fairness metrics such as statistical parity[114] and equalized odds[115] can be adopted. However, these measures do not directly assess counterfactual fairness. Therefore, a counterfactual fairness metric[89] has been proposed, measuring performance discrepancy between factual and counterfactual individuals with different sensitive information. We now provide details on these fairness measures.

**Definition 8** (statistical parity[114]). Statistical parity requires the prediction $\hat{y}$ to be independent with the sensitive attribute $s$, i.e., $\hat{y} \perp s$. Most works are based on a binary attribute and binary classification setting, i.e., $y \in \{0,1\}$ and $s \in \{0,1\}$. In this setting, statistical parity is defined as:

$$P(\hat{y}=1 \mid s=0) = P(\hat{y}=1 \mid s=1). \tag{25}$$

Based on the definition of statistical parity, we can measure statistical parity as:

$$\Delta_{SP} = |P(\hat{y}=1 \mid s=0) - P(\hat{y}=1 \mid s=1)| \tag{26}$$

where a smaller $\Delta_{SP}$ indicates a fairer classifier.

**Definition 9** (equal opportunity[115]). For different subgroups, the equal opportunity demands the probability of an instance being assigned to a positive outcome should be equal, i.e.,

$$P(\hat{y}=1 \mid y=1, s=0) = P(\hat{y}=1 \mid y=1, s=1).$$

Based on the above definition, equal opportunity can be measured by:

$$\Delta_{EO} = |P(\hat{y}=1 \mid y=1, s=0) - P(\hat{y}=1 \mid y=1, s=1)|. \tag{27}$$

Ideally, one also wants to evaluate the quality of the counterfactual fairness directly. However, we lack the ground-truth counterfactual. According to Definition 7, we would expect a model to give similar predictions for factual input and counterfactual input. Thus, Ma et al.[89] introduce a metric $\delta_{CF}$ for graph counterfactual fairness based on the prediction difference of factual and counterfactual, which is defined as:

**Definition 10** (counterfactual fairness metric on graphs[89]). The metric is defined by the discrepancy between the factual prediction and the counterfactual prediction. With a binary sensitive attribute $S$, we set the sensitive attribute with factual value $s$ and counterfactual value $s'$ and get both predictions. Then the difference would be measured as:

$$\delta_{CF} = |P(\hat{y}_{S=s} \mid \mathbf{X}, \mathbf{A}) - P(\hat{y}_{S=s'} \mid \mathbf{X}, \mathbf{A})| \tag{28}$$

where $s, s' \in \{0,1\}^n$ are arbitrary values of sensitive attribute of all nodes. Note that we use $S=s$ and $S=s'$ instead of $S \leftarrow s$ and $S \leftarrow s$ since the desired counterfactual is varied with different counterfactual generation models. $S=s'$ may not be the real counterfactual but it can be generated based on the augmentation method described in Section 3.2.2.

### 3.5 Datasets for graph counterfactual fairness

To evaluate the model performance in terms of fairness, we need graph datasets with both node labels and node sensitive attributes available. In this section, we will list widely-used synthetic dataset and real-world datasets.

#### 3.5.1 Synthetic dataset

Ma et al.[89] propose a synthetic dataset for fair node classification. The sensitive attributes $s_i$, node feature $\mathbf{x}_i$, graph structure $\mathbf{A}$ and labels $y_i$ are generated by:

$$\begin{aligned}
&s_i \sim \text{Bernoulli}(p), \quad \mathbf{z}_i \sim \mathcal{N}(0, \mathrm{I}) \\
&\mathbf{x}_i = \mathcal{S}(\mathbf{z}_i) + s_i \mathbf{v} \\
&P(\mathbf{A}_{ij}=1) = \sigma\big(\cos(\mathbf{z}_i, \mathbf{z}_j) + a \times \mathbb{1}_{[s_i=s_j]}\big) \\
&y_i = \mathcal{B}\Big(\mathbf{w}^\top \mathbf{z}_i + w_s \frac{\sum_{j \in \mathcal{N}_i} s_j}{|\mathcal{N}_i|}\Big)
\end{aligned} \tag{29}$$

where the sensitive attribute $s_i$ is sampled from Bernoulli distribution with $p$ as the probability of $s_i = 1$. The latent representation $\mathbf{z} \in \mathbf{R}^d$ is sampled from a Gaussian distribution. $s_i$ and $\mathbf{z}_i$ work together to generate the node features $\mathbf{X}$, the adjacency matrix $\mathbf{A}$ and the labels $\mathbf{Y}$. $\mathcal{S}(\cdot)$ is a sampling operation which selects $d$ dimensions out of the latent representation and $\mathbf{v} \in \mathbf{R}^d, \mathbf{v} \sim \mathcal{N}(0, \mathrm{I})$ controls influence of sensitive attributes. The sampled latent representation $\mathbf{z}_i$ and $s_i\mathbf{v}$ generate the node feature $\mathbf{x}_i$. $\mathbf{A}$ is generated with the cosine similarity of latent representations and the sensitive attributes. $\mathbb{1}_{[\cdot,\cdot]}$ is the indicator function. It outputs 1 if $s_i = s_j$; otherwise, outputs 0. $a$ is a parameter to control the influence of sensitive attributes. Then the cosine similarity and the output of the indicator function will be fed into a sigmoid function $\sigma(\cdot)$ to compute the probability of each edge. $\mathbf{w} \in \mathbf{R}^d$ is parameters sampled from a normal distribution and $w_s$ is a scalar to control the influence of sensitive attributes on labels. The weighted representation $\mathbf{w}^\top \mathbf{z}_i$ and the weighted average of each node's neighbors' sensitive attribute are used to generate binary value labels $\mathbf{Y}$ with $\mathcal{B}(\cdot)$, where $\mathcal{B}(\cdot)$

maps value into a binary value.

#### 3.5.2 Real-world datasets

There are several real-world datasets widely used to evaluate the performance of fair GNNs.

**1) Pokec-z & Pokec-n**[28]: These two datasets are sampled from a social network dataset Pokec[116]. Pokec is a social network platform in Slovakia and the network has millions of users. Pokec-z and Pokec-n are sampled according to the regions of the users[28]. In these two datasets the region information is the sensitive attribute and the working field of the users is the label for node classification. Pokec-z has 67 797 users and Pokec-n has 66 569 users. The feature dimension for each node in the datasets is 59. Note that the sampling approaches of the datasets vary in different works. Therefore, there may be different statistics in different papers.

**2) UCSD34**[6]: This is a Facebook friendship network of the University of California San Diego. Each node denotes a user, and edges represent the friendship relations between nodes. In this dataset, "gender" is the sensitive attribute, and the task is to predict the major of each user. UCSD34 has 4 132 nodes. The feature dimension for each node is 7.

**3) Credit defaulter**[117]: This dataset contains people′s default payment information[89]. Each node represents an individual and an edge between a pair of nodes represents the similarity of two individuals′ spending and payment patterns. In this dataset, age is the sensitive attribute and the task is to predict the users′ default ways of payment, i.e., credit card or not. This dataset has 30 000 nodes and the feature dimension is 13.

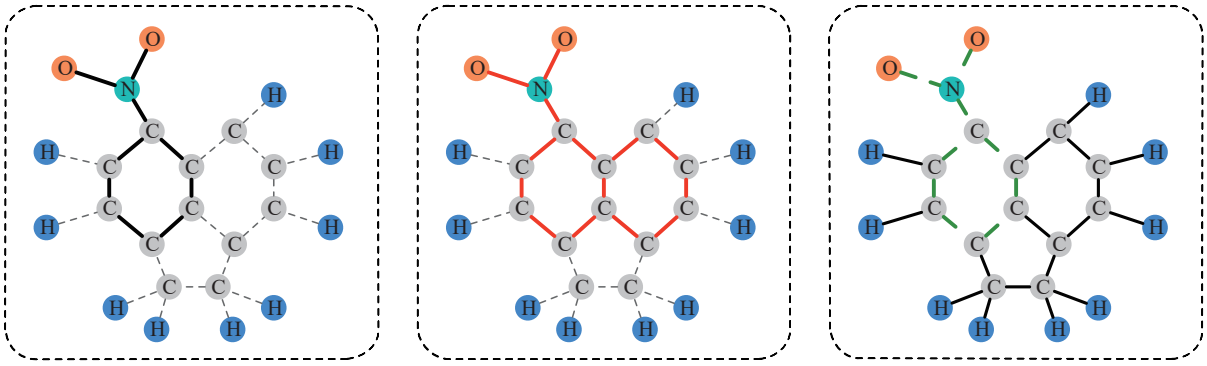
**4) German credit**[118]: It is a client network in a German bank. Each node is a client of the bank and an edge between a pair of nodes denotes the similarity of two clients′ credit accounts. "Gender" is the sensitive attribute and the task is to classify the credit risk of the clients as high or low. This dataset has 1 000 nodes and the feature dimension is 27.

**5) Recidivism (bail)**[119]: It has 18 876 nodes denoting defendants who got released on bail at the U.S. state courts during 1990–2009[89]. Defendants are connected based on the similarity of past criminal records and demographics. In this dataset the defendants′ race is the sensitive attribute. The goal is to classify defendants into bail (i.e., unlikely to commit a violent crime if released) and no bail. This dataset has 18 876 nodes and the feature dimension is 18.

## 4 Counterfactual explanation on graphs

Deep neural networks (DNNs) have achieved remarkable success in various domains, such as molecular biology[120], social networks[9], and financial systems[45]. However, the black-box nature of DNNs hinders their widespread adoption[121]. Transparent and interpretable models are essential to ensure that developers understand model behavior and potential biases, and to gain user trust, especially in high-stakes scenarios[122]. GNNs also face interpretability challenges, which are further exacerbated by complex and discrete graph structures. Thus, improving GNN interpretability is crucial for user trust and further maximizing GNN potential[33, 123–125]. For instance, in disease diagnosis, GNNs might use patient data, genetics, and social connections to predict illness likelihood. Clear explanations can foster trust and understanding among patients and doctors. In drug discovery, GNN interpretations can aid in discovering effective molecular structures[126].

Parallel to the development of explanation techniques on other DNN models, various interpretability methods for GNNs have been developed[127]. Many existing works[46, 127, 128] aim to identify a subgraph that is highly correlated with the prediction result. However, due to the complex graph structure, such methods are likely to get spurious explanations that have a high correlation with the prediction result, instead of the key substructure that causes the label[125]. To avoid the spurious explanation and find the causal explanation that contributes significantly to the prediction, researchers have built various models to get counterfactual explanations on graphs[7, 47, 129–132]. Instead of simply finding a subgraph that is highly correlated with the prediction result, counterfactual explanation on graphs aims to identify the necessary changes to the input graph that can alter the prediction outcome, which can help to filter out spurious explanations. Fig. 4 shows different explanations for a mutagenic prediction result[52, 130]. In the mutagenic prediction task, the Nitrobenzene structure highlighted in black in Fig. 4(a) is the primary cause of mutagenicity, which is the ground-truth factual explanation. The edges highlighted in red in Fig. 4(b) show the explanation obtained by a factual explanation method. Explanation models tend to contain some undesired edges outside the primary cause to give more confident predictions. This is because, in the dataset, edges in red but outside the Nitrobenzene structure having high co-occurrence with the Nitrobenzene structure. Consequently, the models tend to consider these edges as highly correlated with mutagenicity, leading to potentially misleading explanations (spurious information). The green dashed edges in Fig. 4(c) serve as counterfactual explanations for the mutagenicity prediction. The intuition is that removing the edges in the Nitrobenzene structure would likely cause the disappearance of mutagenicity. Thus, counterfactual explanations can help identify the most critical edges for model prediction, aligning well with the ground-truth Nitrobenzene structure[52]. Thus, by focusing on identifying the necessary changes to the input graph that can alter the prediction outcome, counterfactual explanation methods mitigate the impact of spurious explanations and better align with the ground-truth causal factors. Hence, counterfactual explanations on graphs are promising to improve the



(a) Ground-truth explanation (b) Spurious explanation (c) Counterfactual explanation

Fig. 4 Illustration of counterfactual explanation on graphs for mutagenic prediction. (a) is an example of a 5-Nitroacenaphthene molecular structure (factual explanation). (b) is spurious explanation. (c) is the counterfactual explanation of molecular prediction[52]. (Colored figures are available in the online version at https://link.springer.com/journal/11633)

interpretability and trustworthiness of GNNs[33] and many efforts have been taken.

Next, we first introduce background and definition of graph counterfactual explanation. Then we summarize existing works into a general framework of graph counterfactual explanation followed by a detailed review of existing methods[7, 47, 129–138]. Finally, we review widely used metrics and datasets.

## 4.1 Background of graph counterfactual explanations

Various efforts have been taken for the explainability of GNNs, which can be generally categorized into instance-level post-hoc explanations[46, 128], model-level post-hoc explanations[139] and self-explainable methods[29]. Concretely, for each node or graph, instance-level post-hoc explanations aim at finding an important subgraph that leads to the prediction of the target GNN[46]. Many strategies have been proposed to identify key subgraphs, which can be categorized into three groups, i.e., attribution methods[46], decomposition methods[140] and surrogate methods[33, 141]. For example, GNNExplainer[46] is the first work on explaining GNN models. It identifies important subgraph substructure and node attributes that can preserve the prediction of the raw graph. The identified subgraph is treated as an explanation for the input instance. Model-level post-hoc explanations aim to give global-level explanations which are independent of the inputs of the target model[139]. For example, XGNN[139] trains a graph generator to generate graph patterns that maximize a certain prediction of the target model for a model-level explanation. Unlike post-hoc explanations that need another model or strategy to explain a target GNN model, self-explainable approaches aim to learn a GNN model that can simultaneously give predictions and explain predictions[29, 142, 143]. For example, self-explainable graph neural network (SE-GNN)[29] identifies $K$-nearest labeled nodes for each unlabeled node to give explainable node classification.

Although the aforementioned methods[29, 46, 128, 139] can help to understand and explain GNNs, they can be easily stuck at spurious explanations[37]. Specifically, they might only identify the substructures which have a high correlation with the model predictions but cannot distinguish causal effects and spurious effects[125, 144]. As shown in Fig. 4, counterfactual learning can help to find the most critical edges and filter out spurious edges. Generally, for an input instance and a trained GNN model, graph counterfactual explanation aims to explain the prediction by finding a minimal change of the input features and graph structure that would cause the target model to classify the modified input to a desired different class[42]. Hence, most graph counterfactual explanations fall into the instance-level post-hoc category. Following [42], we give the formal definition of instance-level post-hoc graph counterfactual explanation as follows:

**Definition 11** (graph counterfactual explanation). Given a graph neural network classifier $\Phi(\cdot)$ which takes graph (or subgraph) $\mathcal{G}^{\mathrm{F}} = \{\mathcal{V}, \mathcal{E}, \mathbf{X}\}$ as input and gives predictions on graph classification task (or node classification task), a counterfactual explanation $\mathcal{G}^{\mathrm{CF}}$ for the input $\mathcal{G}^{\mathrm{F}}$ is given by solving the following optimization problem:

$$\begin{aligned} &\mathcal{G}^{\mathrm{CF}} = \arg\min_{\mathcal{G}} \operatorname{dist}(\mathcal{G}, \mathcal{G}^{\mathrm{F}}) \\ &\text{s.t. } \Phi(\mathcal{G}) \neq \Phi(\mathcal{G}^{\mathrm{F}}) \text{ and } \Phi(\mathcal{G}) = y_t, \mathcal{G} \in \mathcal{P} \end{aligned} \tag{30}$$

where $y_t$ is a target label for class $t$ so that there will be a counterfactual graph for each class except the original predicted class $\Phi\left(\mathcal{G}^{\mathrm{F}}\right)$. $\operatorname{dist}(\cdot,\cdot)$ measures the distance between $\mathcal{G}$ and $\mathcal{G}^{\mathrm{F}}$, and $\mathcal{P}$ is a set of domain-specific constraints of the obtained counterfactual explanations[145].

This definition says that the graph counterfactual explanation tries to find the counterfactual graph that gives different predictions and also emphasizes that an effective counterfactual graph should be similar to the original factual graph.

Note that counterfactual explanation is different from classical explanation as graph counterfactual explanation can add/delete edges or change feature values instead of simply selecting a substructure highly correlated with the prediction[127].

## 4.2 Methods of graph counterfactual explanations

Different from other explanation methods[46, 128] that only give substructures of high correlation score with the prediction, graph counterfactual explanation aims to get more actionable and useful explanations by understanding how the prediction can be changed in order to achieve an alternative outcome[35]. For example, considering someone who failed to apply for a credit loan, an explanation of why the model makes the decision of rejection is acceptable. However, it is more pragmatic to tell him/her how to adjust a few changeable features or make a few transactions to get approved[146], which is what counterfactual explanations are capable of.

### 4.2.1 General framework of graph counterfactual explanation

Many graph counterfactual explanations methods have been proposed[7, 47, 129–132, 147]. In this subsection, we unify existing methods into a general framework composed of two steps, i.e., graph counterfactual candidate representation step and optimization step. An illustration of the overall framework is shown in Fig. 5, where we use $\bar{y}_i$ to denote the desired label and use $\hat{y}_i$ to denote the original predicted label. As shown in Fig. 5, for a given GNN and the desired label $\bar{y}_i$, we need to generate appropriate masks in input space. Then the perturbed input will be fed into the GNN and give the desired $\bar{y}_i$ instead of the original $\hat{y}_i$. This goal can be achieved by several rounds of optimizations with regularization of input space and predictions (output space). Given the input instance $\mathcal{G}$, the graph counterfactual candidate representation step aims to mathematically represent the actionable space to perturb graph $\mathcal{G}$ for getting counterfactual explanations. We can achieve this goal by considering the decision boundary[129], manipulating the input with masks[47] or any other manipulations[132]. With the counterfactual candidate representation, the optimization step aims to design proper supervision to realize the desiderata and find reasonable graph counterfactual explanations[35]. Next, we give the details.

**Graph counterfactual candidate representation.** Here we focus on post-hoc explanation setting. Given a trained GNN classifier $\Phi(\cdot)$ and a graph/subgraph $\mathcal{G}$, to provide counterfactual explanations on the prediction of $\mathcal{G}$, we need to find counterfactual graph $\mathcal{G}^{\mathrm{CF}}$ that can explain why $\mathcal{G}$ is predicted as label $y$ and what changes we need on $\mathcal{G}$ to result in other predictions. However, in the observational study, we only have access to the observed factual graph $\mathcal{G}$. Thus, the main challenge is how to utilize the observational graph to find a reasonable counterfactual graph $\mathcal{G}^{\mathrm{CF}}$, which has minimal change to $\mathcal{G}$ but a different prediction label from $\mathcal{G}$. A straightforward method is to try all possible masks on the input $\mathbf{A}$ and $\mathbf{X}$ and try to find masks that give the desired prediction. However, there is a combinatory number of candidate masks for an input instance. We need to keep querying the target model to get a prediction for each candidate. Thus the computation cost is unacceptable. Moreover, this straightforward method cannot fulfill the desiderata we mentioned above, such as validity, actionability, sparsity, data manifold closeness and causality[35].

To address these challenges, many graph counterfactual candidate representation approaches have been proposed[7, 47, 129–132]. It should be noted that for other graph explanation methods[46, 128], we only remove edges to find

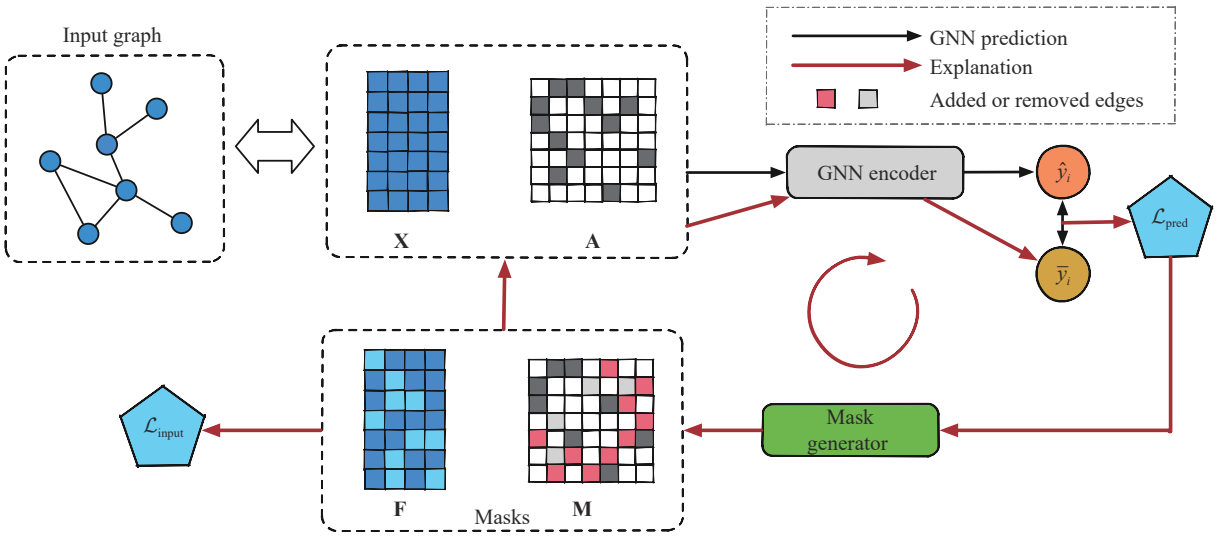



Fig. 5 Overall framework of graph counterfactual explanation (Colored figures are available in the online version at https://link.springer.com/journal/11633)

the critical subgraph as the explanation for the current prediction. However, in counterfactual explanation, we want to find counterfactual graphs by adding and deleting minimal amount of edges that can result in the counterfactual prediction, i.e., the prediction that′s different from $\Phi(\mathcal{G})$ or predictions with a desired label. Therefore, we also need to consider masks on the zeros of the adjacency matrix, which makes the problem more challenging. To mathematically represent the counterfactual graph, we let the supplement of binary adjacency matrix $\mathbf{A}$ as $\overline{\mathbf{A}} = \mathbf{J} - \mathbf{I} - \mathbf{A}$, where $\mathbf{J}$ is an all-one matrix and $\mathbf{I}$ is an identity matrix. Then $\mathbf{C} = \overline{\mathbf{A}} - \mathbf{A}$ is all possible edge perturbation candidates. $\mathbf{C}$ can be decomposed as:

$$\mathbf{C} = \mathbf{C}^- + \mathbf{C}^+ \tag{31}$$

where $\mathbf{C}^- \in \{-1, 0\}^{N\times N}$ and $\mathbf{C}^+ \in \{1, 0\}^{N\times N}$. A negative element of $\mathbf{C}^-$, say $C^-_{ij} = -1$, denotes that the edge $e_{ij}$ exists in $\mathbf{A}$ and can be potentially removed from $\mathbf{A}$. Similarly, a positive element of $\mathbf{C}^+$, say $C^+_{ij} = 1$, denotes that the edge $e_{ij}$ does not exist in $\mathbf{A}$ and can be potentially added[131]. $\mathbf{C} \in \{-1, 0, 1\}^{N\times N}$ and only the diagonal entries of $\mathbf{C}$ are zeros, denoting that we do not add self-loops to nodes. With $\mathbf{C}$, we can select edges from $\mathbf{C}$ with the aid of masks to form counterfactual graph candidates, which can be written as:

$$\mathbf{A}' = \text{Mask}(\mathbf{A}) = \mathbf{A} + \mathbf{C}^- \odot \mathbf{M}^- + \mathbf{C}^+ \odot \mathbf{M}^+ \tag{32}$$

where $\odot$ denotes element-wise product. $\mathbf{M}^- \in \{0, 1\}^{N\times N}$ works with $\mathbf{C}^-$ to select the edges which need to be removed. $\mathbf{M}^-$ maintains zero entries of $\mathbf{C}^-$ and $\mathbf{M}^-_{ij} = 1$ denotes that the edge $e_{ij}$ should be removed from $\mathbf{A}$. $\mathbf{M}^+ \in \{0, 1\}^{N\times N}$. Similarly, $\mathbf{M}^+$ stands for adding edges from the possible candidates. $\mathbf{M}^+$ maintains zero entries of $\mathbf{C}^+$ and $\mathbf{M}^+_{ij} = 1$ denotes that the edge $e_{ij}$ should be added to $\mathbf{A}$. $\mathbf{M} = [\mathbf{M}^-, \mathbf{M}^+]$ summarizes all the adding and deleting operations. Similarly, for node features, a mask $\mathbf{F} \in \{0, 1\}^{N\times d}$ is adopted on the feature matrix $\mathbf{X} \in \mathbf{R}^{N\times d}$ to select important features as:

$$\mathbf{X}' = \text{Mask}(\mathbf{X}) = \mathbf{X} \odot \mathbf{F}. \tag{33}$$

For graph counterfactual candidate representation, we can apply the actionability constraint on the masks, which demands additional domain knowledge. For example, we can not change one′s race to change the college application result. This can be achieved by constraining perturbations in an actionable subset of all potentials, i.e., $\mathbf{F} \in \mathcal{A}_F \in \{0, 1\}^{N\times d}$ and $\mathbf{M} \in \mathcal{A}_M \in \{0, 1\}^{N\times N}$, where $\mathcal{A}_F$ and $\mathcal{A}_M$ are the actionable sets for feature mask and structure mask.

**Objective function.** With the adjacency matrix mask $\mathbf{M}$ and the feature matrix mask $\mathbf{F}$, we have converted the graph counterfactual explanation problem into an optimization problem of finding $\mathbf{M}$ and $\mathbf{F}$ with the constraints in the input space and output space. Specifically, given the target GNN model $\Phi(\cdot)$, the instance $\mathcal{G} = \{\mathbf{A}, \mathbf{X}\}$ and the prediction $\hat{\mathbf{Y}} = \Phi(\mathbf{A}, \mathbf{X})$, we try to find the counterfactual explanation $\mathcal{G}^{CF} = \{\mathbf{A}^{'}, \mathbf{X}^{'}\}$ with $\mathbf{A}^{'} = \text{Mask}(\mathbf{A})$ and $\mathbf{F}^{'} = \text{Mask}(\mathbf{M})$. Based on Definition 11, the objective function of the optimization problem can be written as:

$$\begin{aligned}\mathcal{L}_{\text{explain}}(\mathbf{M}, \mathbf{F}) = \mathcal{L}_{\text{pred}}(\Phi(\mathbf{A}', \mathbf{X}'), \Phi(\mathbf{A}, \mathbf{X})) + \\ \lambda \mathcal{L}_{\text{input}}((\mathbf{A}', \mathbf{X}'), (\mathbf{A}, \mathbf{X}))\end{aligned} \tag{34}$$

where $\lambda$ is the hyper-parameter to balance the contributions of two losses. The above loss functions come from two levels, i.e., the output level and the input level. $\mathcal{L}_{\text{pred}}$ stands for the validity constraint applied on the model prediction, i.e., output level[35]. We utilize $\mathcal{L}_{\text{pred}}$ to make sure $\Phi(\mathbf{A}', \mathbf{X}') \neq \Phi(\mathbf{A}, \mathbf{X})$. An intuitive choice of $\mathcal{L}_{\text{pred}}$ is to maximize the discrepancy between the factual output and counterfactual output. $\mathcal{L}_{\text{input}}$ is used to impose constraints in the input space, which can be a set of loss terms to achieve the goal of sparsity, data manifold closeness and causality. To achieve sparsity, one way is to add a constraint to make factual input and counterfactual input similar, thus the changes from factual to counterfactual are minimized. For example, CF-GNNExplainer[47] adopts a sparsity constraint to minimize the element-wise different between factual input and counterfactual input. For data manifold closeness, a regularization is usually added between counterfactual and factual data distribution to make sure that counterfactual distribution should not violate the factual data distribution by a large margin. As for causality, we need to know the causal relationship of the observed data and make sure the input perturbation follows the causal relationship.

**A unified framework.** Considering the two steps together, the unified framework of graph counterfactual explanation can be written as:

$$\begin{aligned}&\min_{\mathbf{M},\mathbf{F}} \mathcal{L}_{\text{explain}} = \mathcal{L}_{\text{pred}} + \lambda \mathcal{L}_{\text{input}} \\ &\text{s.t.} \quad \mathbf{M} \in \mathcal{A}_C, \quad \mathbf{F} \in \mathcal{A}_F.\end{aligned} \tag{35}$$

The unified framework aims to optimize from both the input level and output level. Based on different choices of graph counterfactual candidate representation approaches and optimization goals, the framework can generalize to different kinds of extensions of counterfactual explanation models. And we can utilize the constraint to achieve various specific goals of counterfactual explanation[35]. Note that the counterfactual explanation should be the necessary change to get different predictions, which corresponds to $\Delta_{\mathbf{A}} = \mathbf{A}' - \mathbf{A}$ and $\Delta_{\mathbf{X}} = \mathbf{X}' - \mathbf{X}$.

#### 4.2.2 Methodologies

Researchers have come up with a series of carefully-

designed models to get counterfactual explanations on graphs[7, 47, 129–132]. They usually differ in terms of $\mathcal{L}_{\text{pred}}$, $\mathcal{L}_{\text{input}}$ and optimization methods. We summarize existing works in Table 5 and show the significant differences between these methods. Next, we will give the details of representative graph counterfactual explanation methods.

**RCExplainer[129].** Targeting at instance-level post-hoc explanation for graph classification task, this work gives a detailed analysis of the decision region of GNNs. The core idea of RCExplainer is to extract the decision region of a target GNN $\Phi(\cdot)$ in the output space of the last convolution layer through linear decision boundary analysis. The decision boundary analysis enables capturing the shared decision logic across multiple input graphs, mitigating the potential overfitting due to the noise present in individual graph instances. As a result, the model generates more robust counterfactual explanations. With the aforementioned analysis, first they leverage a different GNN to perform graph counterfactual candidate representation by generating the edge masks as:

$$\mathbf{Z} = g_\phi(\mathbf{A}, \mathbf{X}), \quad \mathbf{M}_{ij} = f_\theta(\mathbf{z}_i, \mathbf{z}_j) \tag{36}$$

where $g_\phi(\cdot)$ is the GNN parameterized by $\phi$ and $f_\theta(\cdot)$ is an MLP parameterized by $\theta$. Note that RCExplainer only considers removing edges and does not consider adding edges or changing features. With the edge masks, RCExplainer designs the objective function as:

$$\mathcal{L}_{\text{explain}} = \lambda\mathcal{L}_{\text{same}} + (1-\lambda)\mathcal{L}_{\text{opp}} + \beta\mathcal{R}_{\text{sparse}} + \mu\mathcal{R}_{\text{discrete}}$$

where $\mathcal{R}_{\text{sparse}} = \|\mathbf{M}\|_1$ and $\mathcal{R}_{\text{discrete}} = -\frac{1}{|\mathbf{M}|}\sum_{i,j}(\mathbf{M}_{ij}\log(\mathbf{M}_{ij}) + (1-\mathbf{M}_{ij})\log(1-\mathbf{M}_{ij}))$.

The first two terms work for output level, which corresponds to $\mathcal{L}_{\text{pred}}$ in our unified framework. Specifically, $\mathcal{L}_{\text{opp}}$ works for the counterfactual property. It encourages counterfactual graph embedding and the original graph embedding to lie on the opposite side of the decision boundary. $\mathcal{L}_{\text{same}}$ is to ensure that the removed edges themselves should have the same prediction result as the original graph. The last two terms work on the input level, which corresponds to $\mathcal{L}_{\text{input}}$. They encourage the edge mask $\mathbf{M}$ to be sparse, and discrete, i.e., be close to 0 or 1. Bajaj et al.[129] also demonstrate the possibility of extending this work on node classification tasks.

**CF-GNNExplainer[47].** Targeting at instance-level post-hoc explanation for node classification task, CF-GNNExplainer also follows the general framework. Let $\hat{y}_i = \arg\max_c \Phi(\mathbf{A}, \mathbf{X})_i$ be the prediction given by the target GNN $\Phi(\cdot)$ for node $v_i$, where $\Phi(\mathbf{A}, \mathbf{X})_i \in [0,1]^C$ is the predicted class probability distribution for $v_i$ and $C$ is the number of classes. CF-GNNExplainer first defines $\hat{\mathbf{A}} = \mathbf{M} \odot \mathbf{A}$ as the counterfactual example for node $v_i$, where $\mathbf{M}$ is the mask matrix that perturbs $\mathbf{A}$. Then, it uses $\Phi(\cdot)$ to get predictions for counterfactual example as $\hat{y}_i' = \arg\max_c \Phi(\mathbf{M} \odot \mathbf{A}, \mathbf{X})_i$. Note that the parameters of $\Phi(\cdot)$ are fixed and we optimize the mask $\mathbf{M}$ during training. The loss function follows the general framework as $\mathcal{L}_{\text{explain}} = \mathcal{L}_{\text{pred}} + \mathcal{L}_{\text{input}}$. Specifically, $\mathcal{L}_{\text{pred}}$ is given as:

$$\mathcal{L}_{\text{pred}} = -\mathbb{I}_{[\hat{y}_i = \hat{y}_i']} \times l_{\text{NLL}}(\hat{y}_i, \Phi(\mathbf{M} \odot \mathbf{A}, \mathbf{X})_i) \tag{37}$$

Table 5 Summary of graph counterfactual explanation papers. "Model access" is to distinguish between black-box models or not. "Counterfactual" denotes the method to obtain counterfactuals. "Candidate" indicates the modification to obtain counterfactuals, e.g., "M(−, +), F, V" stands for removing and adding edges, feature modification and node modification, respectively. "Factual loss" stands for the classical explanation term, i.e., getting the part that contributes the most to the model prediction. "CF loss" is to ensure the model prediction is changed. For "task", we use G to denote graph classification and use N to denote the node classification.

| Methods | Model access | Counterfactual | Candidate | Factual loss | CF loss | Sparse loss | Task |
|---|---|---|---|---|---|---|---|
| RCExplainer[129] | ✓ | GNN | M(−) | ✓ | ✓ | ✓ | G, N |
| CF$^2$[130] | ✓ | Perturbation | M(−), F | ✓ | ✓ | ✓ | G, N |
| CF-GNNExplainer[146] | ✓ | Perturbation | M(−) | × | ✓ | ✓ | N |
| GNNViz[131] | ✓ | Perturbation | M(−, +) | × | ✓ | ✓ | G |
| Abrate and Bonchi[132] | × | Search | M(−, +) | × | ✓ | ✓ | G |
| CLEAR[133] | × | VAE | M(−, +), F, V | × | ✓ | ✓ | G |
| MEG[7] | ✓ | Perturbation | M(−, +), F, V | × | ✓ | ✓ | G |
| GCFExplainer[134] | × | Random walk | M(−, +), V | × | ✓ | ✓ | G |
| MOO[135] | × | Perturbation | M(−, +), V | ✓ | ✓ | ✓ | N |
| NSEG[136] | ✓ | Neighbor matching | M(−), F | ✓ | ✓ | ✓ | N, G |
| Flow-based[137] | ✓ | ECINN[148] | F | × | ✓ | × | N |
| Ensemble[138] | × | Any | M(−), V | × | ✓ | × | G |
| Chhablani et al.[147] | × | Thresholded Banzhaf value | M(−) | × | × | × | N |

where $\mathbb{I}_{[\hat{y}_i=\hat{y}'_i]}$ is an indicator function which outputs 1 if and only if $\hat{y}_i = \hat{y}'_i$; otherwise it outputs 0. $l_{\text{NLL}}(\cdot,\cdot)$ is the negative log-likelihood loss. Note that this work does not aim at generating a counterfactual with the desired label but just a counterfactual that has a different label from the factual. For the second term $\mathcal{L}_{\text{input}}$, the authors take the element-wise distance between the adjacency matrix and its perturbed counterfactual adjacency matrix as the sparsity constraint. The mask $\mathbf{M}$ in CF-GNNExplainer is a binary perturbation matrix and CF-GNNExplainer iteratively optimizes the mask $\mathbf{M}$ with the regularization of $\mathcal{L}_{\text{input}}$ and $\mathcal{L}_{\text{pred}}$. Finally, they retrieve the optimal counterfactual explanation as the difference between the original adjacency matrix $\mathbf{A}$ and the optimal counterfactual adjacency matrix $\mathbf{M}^* \odot \mathbf{A}$. $\Delta\mathbf{A}^* = \mathbf{A} - \mathbf{M}^* \odot \mathbf{A}$ gives the minimal change needed to change model prediction on the input instance.

**CF$^2$[130].** Most of the graph explanation techniques focus on performing factual reasoning to find a subgraph that is highly correlated with the prediction. And the aforementioned graph counterfactual explanation methods only aim at finding counterfactual explanations which can change the prediction result. CF$^2$ combines these two approaches and aims to get necessary and sufficient explanations. Here sufficiency means for factual reasoning the information induced by explanation should be enough to produce the same prediction as the original graph. Necessity means that removing the minimal part will result in different prediction results. In this work, same as the graph counterfactual candidate representation step, CF$^2$ first defines masks on the adjacency matrix and feature matrix as $\mathbf{M}$ and $\mathbf{F}$. It tackles graph classification task, where each graph $\mathcal{G}_k = \{\mathbf{A}_k, \mathbf{X}_k\}$ is associated with a ground-truth class label $y_k \in \mathcal{C}$ with $\mathcal{C} = \{1, 2, \cdots, r\}$ as the set of classes. Let $\hat{y}_k = \arg\max_{c\in\mathcal{C}} P_\Phi(c \mid \mathbf{A}_k, X_k)$ be the predicted label made by model $\Phi$ for $\mathcal{G}_k$. We set $S_f(\mathbf{M},\mathbf{F}) = P_\Phi(\hat{y}_k \mid \mathbf{A}_k \odot \mathbf{M}_k, \mathbf{X}_k \odot \mathbf{F}_k)$ and $S_c(\mathbf{M},\mathbf{F}) = -P_\Phi(\hat{y}_k \mid \mathbf{A}_k - \mathbf{A}_k \odot \mathbf{M}_k, \mathbf{X}_k - \mathbf{X}_k \odot \mathbf{F}_k)$. Then the condition for factual reasoning and counterfactual reasoning can be given as[130]:

$$\begin{aligned}\mathcal{L}_f = {} & \text{ReLU}\big(\gamma + P_\Phi(\hat{y}_{k,s} \mid \mathbf{A}_k \odot \mathbf{M}_k, \mathbf{X}_k \odot \mathbf{F}_k) - \\ & S_f(\mathbf{M}_k, \mathbf{F}_k)\big) \\ \mathcal{L}_c = {} & \text{ReLU}\big(\gamma - S_c(\mathbf{M}_k, \mathbf{F}_k) - \\ & P_\Phi(\hat{y}_{k,s} \mid \mathbf{A}_k - \mathbf{A}_k \odot \mathbf{M}_k, \mathbf{X}_k - \mathbf{X}_k \odot \mathbf{F}_k)\big)\end{aligned} \tag{38}$$

where $\gamma$ is the parameter to control the margin. $S_f(\cdot)$ and $S_c(\cdot)$ control the factual reasoning and counterfactual reasoning, respectively. $\hat{y}_{k,s}$ denotes the label other than $\hat{y}_k$ that has the largest probability score predicted by the GNN model. $\mathcal{L}_f$ aims to ensure that the predicted label $\hat{y}_k$ has the highest probability among all labels when only using the factual explanation sub-graph $\mathbf{A}_k \odot \mathbf{M}_k$ and $\mathbf{X}_k \odot \mathbf{F}_k$, which guarantees that the prediction remains the same for factual explanation. On the other hand, $\mathcal{L}_c$ is to ensure that when the factual explanation subgraph is removed, the probability of $\hat{y}_k$ should be lower than the probability of $\hat{y}_{k,s}$ by $\gamma$, leading to a change in the prediction. Therefore, $\mathcal{L}_f$ and $\mathcal{L}_c$ serve to ensure that the subgraph explanation is both necessary and sufficient for the model's prediction. With these two conditions, the loss function of CF$^2$ is:

$$\mathcal{L}_{\text{explain}} = \mathcal{L}_{\text{input}} + \lambda(\alpha\mathcal{L}_f + (1-\alpha)\mathcal{L}_c) \tag{39}$$

where $\mathcal{L}_{\text{input}} = \|\mathbf{M}_k\|_1 + \|\mathbf{F}_k\|_1$ works on the input level to make the masks sparse.

**GNNViz[131].** GNNViz only considers the perturbation of adding and removing edges, i.e., the perturbed adjacency matrix is $\mathbf{A}' = \mathbf{A} + \mathbf{C}^- \odot \mathbf{M}^- + \mathbf{C}^+ \odot \mathbf{M}^+$, where $\mathbf{M} = [\mathbf{M}^-, \mathbf{M}^+]$ is the mask matrix. To generate a counterfactual explanation, GNNViz encourages the target model to have higher confidence in other labels than the origin predicted label, i.e.,

$$\mathcal{L}_{\text{pred}}(\mathbf{M}) = -\min\left\{\max_{t\notin\Omega} p\left(t|\mathbf{A}', \mathbf{X}\right) - \max_{c\in\Omega} p\left(c|\mathbf{A}', \mathbf{X}\right), \kappa\right\} \tag{40}$$

where $\Omega$ is the set of original predicted labels, and $\kappa$ is the confidence gap used to encourage the model to give more confidence on changed labels. The final loss function is

$$\min_{\mathbf{M}} \mathcal{L}_{\text{pred}}(\mathbf{M}) - \lambda_1\|\mathbf{M}^-\|_1 + \lambda_2\|\mathbf{M}^+\|_1 \tag{41}$$

where $\lambda_1$ and $\lambda_2$ are scalars to control the contribution of two regularization terms. $-\lambda_1\|\mathbf{M}^-\|_1 + \lambda_2\|\mathbf{M}^+\|_1$ is used to get a sparse counterfactual adjacency matrix, i.e., by minimizing $-\lambda_1\|\mathbf{M}^-\|_1$, we encourage removing more edges and by minimizing $\lambda_2\|\mathbf{M}^+\|_1$, we prefer adding fewer edges.

**MEG[7].** Given a target model for molecular property prediction (graph classification), MEG aims to get counterfactual explanation for the prediction of each molecule, which can help to find molecules that have the desired properties. Our unified framework gives a feasible answer to counterfactual explanation questions, i.e., first find a way to represent the potential manipulation space for generating counterfactuals, then design objectives on output level and input level to make the model give different prediction results with reasonable regularization. The framework is flexible and can be easily adopted in different frameworks, e.g., the reinforcement learning framework in MEG. In the reinforcement learning framework of MEG, it defines the action space as:

$$\mathcal{A}_t = \mathcal{A}_a \cup \mathcal{A}_b^+ \cup \mathcal{A}_b^- \cup \{\perp\} \tag{42}$$

where $\mathcal{A}_a$ stands for adding nodes (atoms), $\mathcal{A}_b^+$ or $\mathcal{A}_b^-$ stands for adding or removing edges (bonds), $\perp$ means no

action is taken. The reward function of MEG is given as

$$\mathcal{R}_{\text{explain}} = \alpha\mathcal{R}_{\text{pred}} + (1-\alpha)\mathcal{R}_{\text{input}} = \\ -\alpha p(c|\mathcal{G}^{\text{CF}}) + (1-\alpha)\mathcal{K}(\mathcal{G}, \mathcal{G}^{\text{CF}}) \quad (43)$$

where $c$ is the predicted label of the original molecule graph $\mathcal{G}$. $\mathcal{G}^{\text{CF}}$ is the generated counterfactual graph. $\mathcal{K}(\cdot,\cdot)$ is defined as the similarity score of the representations of $\mathcal{G}$ and $\mathcal{G}^{\text{CF}}$. $\alpha$ controls the contribution of $\mathcal{L}_{\text{pred}}$ and $\mathcal{L}_{\text{input}}$. The loss function encourages the model to give less confidence to the original predicted label and have minimal changes to generate counterfactual graph $\mathcal{G}^{\text{CF}}$.

**CLEAR[133].** CLEAR is another method that aligns well with our unified framework, with a focus on graph classification. Unlike other methods that use masks to perturb the graph structure, CLEAR uses a variational auto-encoder (VAE) to generate the counterfactual graph. The advantage of using VAE is that, by properly injecting auxiliary information, VAE can preserve the latent causal structure of the graph. The counterfactual graph is generated as:

$$\mathcal{G}^{\text{CF}} = \{\mathbf{A}^{\text{CF}}, \mathbf{X}^{\text{CF}}\} = \text{VAE}(\mathcal{G}, y_t, s) \quad (44)$$

where $y_t$ is the desired label for $\mathcal{G}^{\text{CF}}$ and $s$ is an auxiliary observed attribute, such as an index for each sample or class label. CLEAR formulates the loss function as:

$$\mathcal{L}_{\text{pred}} = \alpha \times l(\Phi(\mathcal{G}^{\text{CF}}), y_t) + \mathcal{L}_{\text{VI}} \\ \mathcal{L}_{\text{input}} = \|\mathbf{A}^{\text{CF}} - \mathbf{A}\|_F^2 + \|\mathbf{X}^{\text{CF}} - \mathbf{X}\|_F^2 \quad (45)$$

where $\Phi(\cdot)$ is the target GNN, $y_t$ is the desired label for $\mathcal{G}^{\text{CF}}$, $\mathcal{L}_{\text{VI}}$ is defined as the variational distribution regularization for VAE and $l(\cdot,\cdot)$ is the cross-entropy prediction loss. The regularization term $\|\mathbf{A}^{\text{CF}} - \mathbf{A}\|_F^2 + \|\mathbf{X}^{\text{CF}} - \mathbf{X}\|_F^2$ is applied to both the adjacency matrix and feature matrix to ensure that the model makes minimal changes to the graph structure, i.e., adding or removing fewer edges and modifying fewer features. This regularization penalizes large changes in the graph, promoting explanations that are as close as possible to the original graph while still achieving the desired counterfactual effect, i.e., making $\Phi(\mathcal{G}^{\text{CF}})$ to predict the label $y_t$.

**Brain[132].** In the context of brain network classification, Abrate and Bonchi[132] propose a search-based counterfactual explanation approach to produce local post-hoc explanations of any black-box graph classifier.

The approach is composed of two steps. In the first step, starting with the original graph, the authors iteratively query the target model by adding or removing several edges to the original graph. The process continues until the prediction changes. This step allows the authors to identify an appropriate number of edge changes required to alter the prediction result. In the second step, the authors iteratively add or remove edges to create a counterfactual graph that has a smaller edit distance from the original input. The authors' two-step framework aligns with our unified framework where the search space is used to obtain graph counterfactual candidates, and the first and second steps play similar roles to the objective function, i.e., altering the prediction and making smaller changes to the inputs.

**Other works.** In this part, we will explore a variety of studies that, while demonstrating strong alignment with the unified framework outlined in the previous section, exhibit distinct emphases on their respective goals or methodologies. For each work, we do not introduce too many technical details but aim to highlight the diversity of these works. These works are also shown in Table 5.

**1) Model-level graph counterfactual explanation.** The aforementioned works focus on instance-level explanations, i.e., explaining the predictions made for specific instances. In contrast, GCFExplainer[134] aims to provide model-level counterfactual explanations for GNNs. In graph classification tasks, given the target GNN classifier $\Phi(\cdot)$, the global counterfactual explanation aims to find a global rule $r_c(\cdot)$ for each class $c$ such that for each graph $\mathcal{G}_i$ with $\Phi(\mathcal{G}_i) \neq c$, $r_c(\cdot)$ can modify $\mathcal{G}_i$ to counterfactual graph $r(\mathcal{G}_i)$ with prediction $\Phi(r(\mathcal{G}_i)) = c$. In GCFExplainer, Huang et al.[134] simplify this problem of finding $r(\cdot)$ into finding a set of representative counterfactual graphs for each class, where counterfactual graph means modified from the original graph but with a different label with the original graph. Based on the language of our framework, in the graph counterfactual candidate representation step, they structure the search space for counterfactuals. The loss function incorporates regularization at both the input level and output level, where the regularization encourages the model prediction in the desired class at the output level and regularization ensures the manipulation should be sparse at the input level. Those representative counterfactual graphs are treated as model-level counterfactual explanations.

**2) Factual and counterfactual explanation.** For graph classification task, factual explanation aims to identify the key subgraph of $\mathcal{G}_i$ whose information is sufficient to preserve the label information of $\mathcal{G}_i$. Counterfactual explanation aims to find minimal manipulation of $\mathcal{G}_i$ that is necessary to result in different prediction. CF$^2$[130] is an early effort to combine these two directions to get a sufficient and necessary explanation for target GNN models as discussed in Section 4.2.2. MOO[135] is another work combining these two directions and trying to find optimal explanations that are well-balanced between sufficiency and necessity. MOO also aligns well with our framework, where the perturbation is employed to get graph counterfactual candidates and both regularizations of sufficiency and necessity are applied as the objective.

However, Cai et al.[136] argue that the trade-off of necessity and sufficiency in these approaches is heuristically determined. Hence, the obtained explanation might not be an optimal balance for both desiderata. NSEG[136] is proposed to alleviate the issue. NSEG adopts a technique named probability of necessity and sufficiency (PNS) to quantify the necessity and sufficiency and aims to maximize the lower bound of PNS to find the optimal explanation. Specifically, NSEG hinges on a nearest neighbor matching strategy to obtain graph counterfactual candidates and then uses counterfactuals to maximize the lower bound of NSEG.

**3) Generative model-based explanation.** Another intriguing development in the realm of graph counterfactual explanation is the generative model-based counterfactual explanations[133, 137]. These works also align with our unified framework, where they use generative models to obtain graph counterfactual candidates and adopt regularizations on output and input level to supervise the learning process. For example, CLEAR[133] uses VAE to generate counterfactual graphs. Ohly[137] designs a flow-based model to generate counterfactuals. Specifically, Ohly[137] adopts efficient counterfactuals from invertible neural networks (ECINN)[148], a normalizing-flow-based generative model that can generate counterfactual candidates, to alter the node features. A classifier is adopted to make sure that the counterfactual examples get changed prediction results compared with original predictions.

**4) Ensemble-based explanation.** Instead of sticking to one single model, Prado-Romero et al.[138] introduce an ensemble-based approach, which leverages the combined power of multiple models to improve the overall interpretability and performance. In their model, multiple graph counterfactual explanation models are used to generate a set of counterfactual explanations on graphs. Then the shared substructure of these explanations is treated as the final counterfactual explanation.

**5) Semivalue-based explanation.** Different from previous methods that involve a training process, Chhablani et al.[147] introduce a non-learning-based approach based on semivalues[149]. Specifically, they design a function to measure the importance of edges from the utility functions in cooperative game theory based on Banzhaf values[150].

## 4.3 Taxonomy of graph counterfactual explanation

In this subsection, we categorize existing graph counterfactual explanation methods from four perspectives: counterfactual graph, objective, performed task and model access. A summary of the categorization can be found in Table 5. Next, we will introduce each perspective in detail.

**1) Counterfactual graph.** From the perspective of counterfactual graph design, existing methods can be classified into perturbation-based methods[7, 47, 130, 131, 135], search-based methods[132], neural network-based methods[129, 133, 137], and other methods[134, 136, 147]. Perturbation-based methods develop perturbation masks for the adjacency matrix and feature matrix, which convert counterfactual graph generation into mask optimization problem[7, 47, 130, 131, 135]. These methods are easy to implement and can be applied to explain a variety of target models. Search-based method employs a straightforward approach, i.e., iteratively removing or adding edges until the desired requirements are met[132]. It needs to query the target model but does not require any additional information. Neural network-based methods utilize various neural networks, such as GNN[12], VAE[151] and ECINN[148], to learn the counterfactual graph structure. Neural networks possess a powerful representation capacity to capture the latent relationships among variables, providing semantically meaningful counterfactuals. Other methods obtain the counterfactual graphs with different specific concerns. For example, random walk is used in [134] to help identify important representative nodes for graphs in a class, where the shared nodes are thought to be representative for the class. Then these nodes are used to form counterfactual graphs, which work as the explanation of the mentioned class. Prado-Romero et al.[138] adopt ensemble learning to combine explanations from multiple graph counterfactual models. Chhablani et al.[147] utilize a function to measure the importance of edges to decide what edges to delete.

**2) Objective.** The graph counterfactual candidate representation step formulates the search space for counterfactual graphs and generates a selection of counterfactual graph candidates. Following this, objectives are applied to assess the model′s performance concerning various factors. Within the context of graph counterfactual explanation, objectives can take on different forms, including reward functions, loss functions, and termination conditions for the search process. Regardless of the form, we can categorize the objectives′ desiderata into several types, such as factual regularization, counterfactual regularization, and sparsity regularization. Most of the graph counterfactual explanation methods incorporate counterfactual regularization, which requires changes in model predictions[7, 47, 129–138]. The regularization usually involves decreasing the confidence score of the original prediction or ensuring the counterfactual prediction lies on the opposite side of the decision boundary. With this regularization, the model′s prediction is modified, allowing for the examination of the desired change. Some methods also employ factual regularization, which assists in identifying sufficient information to complement counterfactual regularization[129, 130, 135, 136]. These methods aim to identify important substructures from two perspectives and they are capable of more precisely capturing the important substructure. By using the substructure, we can maintain the original prediction while removing them res-

ulting in significant changes in the model's prediction. They usually employ a regularization to increase the prediction score when solely using the explanation substructure. For the sparsity regularization, it is employed to promote concise and interpretable explanations by limiting the number of edges or feature changes in the counterfactual explanation[7, 47, 129–136]. The regularization minimizes the $l_1$ norm of the masks of counterfactual explanation or the difference between the counterfactual graph and the original graph.

**3) Tasks.** Regarding the tasks performed, existing works can be generally categorized into graph-level and node-level tasks. In graph-level tasks, each graph, considered as an instance, is independent and identically distributed (i.i.d.). The majority of existing methods aim to find reasonable counterfactual explanations for a single prediction result in graph classification tasks[7, 129–134, 136, 138]. Among these works, the objective of GCFExplainer[134] is slightly different, as it seeks to find several representative counterfactual graphs for model-level explanations. Some studies focus on the node classification setting[47, 129, 130, 135–137, 147], which entails finding reasonable modifications that can result in changes to the node classification outcomes. These methods aim to identify and understand the impact of specific modifications on the node classification results within the graph structure.

**4) Model access.** In the context of explainers, it is essential to discuss the degree of access to the target model, particularly whether it is treated as black box or not. Black-box setting means internal workings of the target model are hidden or unknown, making it challenging to explain predictions of the target model. Explanation techniques for black-box setting[132–135, 138, 147] often involve querying the black box model and collecting its feedback to guide their neural networks[133], masks learning[135], or search processes[132]. For white-box setting which allows access to the target model's parameters, the problem is considerably easier. These methods[7, 47, 129–131, 136, 137] can leverage the target model's prediction probabilities[131], embeddings[129], or even model parameters[47] to supervise the explainer. With these additional information, explainers with access to the target model can more effectively generate insightful and accurate explanations of the target model's decision-making process.

## 4.4 Evaluation metrics for graph counterfactual explanations

Many quantifiable proxies have been proposed to measure the performance of counterfactual explanations, such as accuracy, fidelity, sparsity, stability and graph edit distance. GRETEL also discusses the evaluation of graph counterfactual explanation[146].

**Accuracy, F1 and AUC.** When ground-truth rationales are available for graphs, a direct evaluation method can be employed by comparing the identified explanatory components to the ground-truth explanations[128]. This allows for a comprehensive assessment of the explanation's quality and relevance. Metrics such as accuracy score, F1 score and ROC-AUC score can be computed based on this comparison to the ground-truth explanation. Higher scores signify more accurate explanations.

**Fidelity.** When ground-truth explanations are unavailable, fidelity-based metrics can be employed to measure the explanation performance. The underlying idea is that explanatory substructures should have a more significant impact on predictions, thus removing them or adding them will have a noticeable change in the GNN prediction score. Fidelity+[127] is computed by removing all input elements first, then gradually adding edges with the highest explanation scores. Intuitively, a faster increase in GNN's prediction indicates stronger fidelity of obtained explanations. On the contrary, Fidelity-[127] is computed by sequentially removing edges following assigned importance weight. A faster performance drop represents stronger fidelity of removed explanations.

**Sparsity.** Sparsity, in the context of graph counterfactual explanation, refers to the minimal number of changes or interventions required to generate a counterfactual instance that would lead to a different outcome[129]. A sparse explanation is desirable because it provides an understandable and concise rationale for the change in the outcome. In other words, the fewer changes needed, the easier it is for humans to comprehend and trust the explanation. Graph counterfactual explanations involve generating an alternative graph with a limited number of modifications to original graph, such as adding or removing edges or nodes. These changes should be sufficient to alter the outcome predicted by the model. Sparsity aims to minimize these modifications while maintaining the desired counterfactual effect.

**Stability.** Ideally, good explanations should capture the intrinsic causal connections between input graphs and their labels. This criterion necessitates that the identified explanations remain stable under small perturbations[130]. By making perturbations in the input graph, such as adding new nodes or removing edges, the stability scores can be calculated by comparing changes in explanations before and after the perturbation.

**Graph edit distance (GED).** The graph edit distance (GED)[52, 146] is employed to measure the structural distance between the original graph $\mathcal{G}^{\mathrm{F}}$ and the counterfactual graph $\mathcal{G}^{\mathrm{CF}}$. Let $p_i \in \mathcal{P}(\mathcal{G}^{\mathrm{F}}, \mathcal{G}^{\mathrm{CF}})$ represent a path consisting of actions to be performed on $\mathcal{G}^{\mathrm{F}}$ in order to generate $\mathcal{G}^{\mathrm{CF}}$. Each path contains specific actions $r_j \in p_i$ with an associated cost $\gamma(r_j)$. The GED can be defined as $\mathrm{GED} = \min_{p_i \in \mathcal{P}(\mathcal{G}^{\mathrm{F}}, \mathcal{G}^{\mathrm{CF}})} \sum r_j \in p_i \gamma(r_j)$. Ideally, a smaller GED is preferred, indicating that $\mathcal{G}^{\mathrm{F}}$ is close to $\mathcal{G}^{\mathrm{CF}}$. This implies that only minimal changes are required in the input space to alter the prediction result.

## 4.5 Datasets for counterfactual explanations on graphs

To evaluate the performance of graph counterfactual explanation models on both node classification tasks and graph classification tasks, a set of synthetic datasets and real-world datasets are used by researchers.

In this section, we will first introduce the design of synthetic datasets. Then we will discuss the widely-used real-world datasets.

### 4.5.1 Synthetic datasets

Different from images and texts, it is typically difficult to get the ground truth explanations for real-world graphs due to the complex graph structure. To address this problem, we can develop synthetic graphs with groundtruth explanations. Next, we introduce some widely used synthetic datasets for explainable node classification and graph classification.

**1) BA-shapes**[46]: It is a single graph dataset which consists of a base Barabasi-Albert (BA) graph (300 nodes) and 80 "house"-structured motifs (each for 5 nodes). The motifs are randomly attached to the base BA graph. Nodes in the base BA graph are labeled as 0 while the nodes in motifs are labeled as 1, 2 and 3 based on their location in the "house". The ground truth explanation for each node is the corresponding motif.

**2) Tree-cycles**[46]: It is a single graph with an 8-layer balanced binary tree as the base graph and 80 cycle motifs (each for 6 nodes) are randomly attached to the base graph. Similar to the BA-shapes dataset, nodes in the base graph are labeled as 0 and those in the motifs are labeled as 1. The ground truth explanation for each node is the corresponding cycle.

**3) Tree-grid**[46]: This dataset is the same as tree-cycles except that 3-by-3 grid motifs are randomly attached to the base tree graph to replace the cycle motifs.

**4) BA-2motifs**[128]: It is a graph classification dataset which contains 800 graphs. "House" motifs are attached to half of the graphs while "cycle" motifs are attached to the other half of the graphs. And the motif works as the ground-truth explanation.

### 4.5.2 Real-world datasets

In addition to the synthetic graphs, there are some real-world graphs which are used for evaluating graph counterfactual explanation performance.

**1) Mutagenicity**[152]: This is a graph classification dataset where each graph corresponds to a molecule. Nodes represent atoms and edges represent chemical bonds. Molecules are labeled with consideration of their chemical properties, and discriminative chemical groups are identified using prior domain knowledge.

**2) NCI1**[153]: This is a graph classification dataset from the cheminformatics domain, where each graph is a chemical compound similar to mutagenicity dataset. It is relative to anti-cancer screens where the chemicals are assessed as positive or negative for cell lung cancer.

**3) TOX21**[154]: It is a dataset comprised of 12 060 training samples and 647 test samples that represent chemical compounds. There are 801 "dense features" that represent chemical descriptors, such as molecular weight, solubility or surface area, and 272 776 "sparse features" that represent chemical substructures. For each sample, there are 12 binary labels that represent the outcome (active/inactive) of 12 different toxicological experiments.

**4) ESOL**[154]: This is a regressive task dataset on the water solubility of chemical compounds, which includes 1 129 compounds. It can be used to estimate solubility directly from chemical structure.

**5) Autism spectrum disorder (ASD)**[155]: It is a publicly-available dataset for human-brain. Abrate and Bonchi[132] only focus on the portion of a dataset containing children below 9 years of age, which are 49 individuals in the condition group, labeled as autism spectrum disorder (ASD) and 52 individuals in the control group, labeled as typically developed (TD).

**6) Attention deficit hyperactivity disorder (ADHD)**[156]: It is also a publicly-available dataset for human-brain. This dataset contains 190 individuals in the condition group, labeled as ADHD and 330 individuals in the control group, labeled as TD.

**7) CiteSeer**[12]: This is a citation network where nodes denote papers and citations between two papers are represented as edges. The nodes have six classes which stand for different categories.

# 5 Counterfactual link prediction and recommendation

In the above sections, we mainly focus on counterfactual fairness and counterfactual explanation of GNNs for node classification and graph classification. Link prediction[157], which aims to predict the missing links in a graph, is another important graph mining task. It has wide adoptions on various applications such as recommender systems[62], knowledge graphs[4] and social networks[6]. Recently, counterfactual link prediction[30] was investigated, which aims to explore the root causes of the formation of links, filtering out the spurious factors. Recommender systems, as a special case of link prediction task, can also benefit from removing spurious information and relying on causal information. Hence, counterfactual learning is attracting increasing attention in link prediction task[30] and recommender systems[158–162]. In this section, we will give a comprehensive review of existing works on counterfactual link prediction and counterfactual recommendation with graph learning.

## 5.1 Counterfactual link prediction

We summarize discussed works in Table 6. In this subsection, we will first introduce background knowledge about link prediction and then focus on the very recent

Table 6 Summary of counterfactual link prediction and counterfactual recommendation

| Methods | Categories |
|---|---|
| CFLP[30] | Link prediction |
| KGCF[163] | Knowledge graph completion |
| CR[158] | Recommendation |
| UKGC[161] | Recommendation |
| CGKR[160] | Recommendation |
| COCO-SBRS[162] | Recommendation |
| FairGap[164] | Recommendation |
| Boratto et al.[165] | Recommendation and explanation |
| GREASE[159] | Recommendation and explanation |

counterfactual link prediction work.

**Background.** Link prediction is widely used in applications like social recommendation[166], knowledge graph completion[167], and chemical interaction prediction[168]. Approaches to link prediction can be categorized into heuristic-based[169, 170] and representation learning-based methods[23, 171]. Heuristic-based approaches rely on statistical properties to estimate link likelihoods[172]. For instance, the common-neighbor index scores node pairs based on shared neighbors[172]. However, these approaches only utilize topology information, neglect node features, and may make overly strong assumptions, limiting their broader applicability[23]. Representation learning-based approaches predict link probabilities by learning node representations and computing dot products between them. For example, VGAE[171] learns meaningful low-dimensional representations that can reconstruct the original graph structure and feature information. Then, they use the representations of a pair of nodes $(v_i, v_j)$ to predict the probability of link existence. WalkPool[23] adopts a learnable random-walk-based mechanism to encode both node features and graph topology in the node representations.

**Motivation of counterfactual link prediction.** Counterfactual link prediction (CLP) lies in the intersection of counterfactual learning and link prediction, which aims to remove spurious factors and investigate the root causes of the link formation[30]. For example, assume that Alice and Adam are friends because they live in the same community and they share some same interests. If we want to dig out how likely two people with the same interests will become friends, the community information can be a spurious factor and we need to mitigate the influence of communities. CLP aims to know the real causes from observational data instead of performing impossible experiments to have Alice and Adam grow up in different communities and see if they will become friends. CFLP[30] is the only work on counterfactual link prediction until now, which aims to know the existence of links without the influence of community information, i.e., "would the link exist if the graph structure became different from observation?".

**Methodologies.** Given the observed link distribution, the core idea of CFLP is to model the counterfactual link distribution where the community factor has been changed, which corresponds to having Alice and Adam grow up in different communities. Specifically, CFLP is composed of two steps: 1) obtain counterfactual links; 2) train informative node representations to predict both factual link distribution and counterfactual link distribution.

Since the community information is unobserved, CFLP first utilizes Louvain[173] to perform community detection based on the graph structure. With the detected communities, it constructs the treatment matrix $\mathbf{T}$ with $\mathbf{T}_{ij} = 1$ if $e_{ij}$ is an intra-community edge and $\mathbf{T}_{ij} = 0$ otherwise.

One challenge is to induce the counterfactual links, i.e., how to know the relationship if Alice and Adam grow up in different communities. CFLP proposes to find another two people who are similar to Alice and Adam but grow up in different communities and treat their link status as the counterfactual link for Alice and Adam. Specifically, for each pair of nodes $(v_i, v_j)$, CFLP finds a pair of nodes $(v_a, v_b)$ that is similar with $(v_i, v_j)$ but has different treatment $\mathbf{T}_{ab} = 1 - \mathbf{T}_{ij}$ as:

$$(v_a, v_b) = \underset{(v_a, v_b) \in \mathcal{V}}{\arg\min} \left\{ h\left( (v_i, v_j), (v_a, v_b) \right) \mid \mathbf{T}_{ab} = 1 - \mathbf{T}_{ij} \right\} \tag{46}$$

where $\mathcal{V}$ is the set of nodes in the graph, $h(\cdot, \cdot)$ is a metric for measuring the distance between two pairs of node pairs. However, the computation cost of computing the distance with respect to both graph topology and feature information for all pairs of node pairs is intractable. To resolve this issue, CFLP uses the distance of node embeddings to simplify the problem:

$$\begin{aligned}(v_a, v_b) = \underset{v_a, v_b \in \mathcal{V}}{\arg\min} \Big\{ & d\left(\tilde{\mathbf{x}}_i, \tilde{\mathbf{x}}_a\right) + d\left(\tilde{\mathbf{x}}_j, \tilde{\mathbf{x}}_b\right) \mid \\ & \mathbf{T}_{ab} = 1 - \mathbf{T}_{ij}, d\left(\tilde{\mathbf{x}}_i, \tilde{\mathbf{x}}_a\right) + d\left(\tilde{\mathbf{x}}_j, \tilde{\mathbf{x}}_b\right) < 2\gamma \Big\}\end{aligned} \tag{47}$$

where $\tilde{\mathbf{x}}_i$ is the node embedding of node $v_i$ obtained from the encoder MVGRL[174], $d(\cdot, \cdot)$ is specified as the Euclidean distance in the embedding space, and $\gamma$ is a hyper-parameter that defines the minimal distance that two nodes are considered similar. Here, we utilize $\tilde{\mathbf{x}}_i$ instead of $\mathbf{z}_i$ to denote the node embedding since there is a different encoder to learn the embedding $\mathbf{z}_i$ for downstream link prediction task in CFLP. With this simplification, the model only needs $O(N^2)$ comparisons to find the counterfactual node pairs. Then, the counterfactual treatment matrix $\mathbf{T}^{CF}$ and the counterfactual adjacency matrix $\mathbf{A}^{CF}$ can be obtained as:

$$\mathbf{T}_{ij}^{CF}, \mathbf{A}_{ij}^{CF} = 1 - \mathbf{T}_{ij}, \mathbf{A}_{ab}, \\ \text{if } \exists\,(v_a, v_b) \in \mathcal{V} \times \mathcal{V} \text{ satisfies (47).} \tag{48}$$

The second step is to train a representation learning model based on both factual links and counterfactual links. Following the commonly-used link prediction scheme, CFLP utilizes a GCN encoder to get node representations $\mathbf{Z}$ as $\mathbf{Z} = \text{GCN}(\mathbf{A}, \mathbf{X})$. Then an MLP decoder is adopted to reconstruct both factual links and counterfactual links as:

$$\widehat{\mathbf{A}}_{ij} = \text{MLP}\left([\mathbf{z}_i \odot \mathbf{z}_j, \mathbf{T}_{ij}]\right) \\ \widehat{\mathbf{A}}_{ij}^{CF} = \text{MLP}\left([\mathbf{z}_i \odot \mathbf{z}_j, \mathbf{T}_{ij}^{CF}]\right) \tag{49}$$

where $\mathbf{z}_i$ is the node representation of $v_i$ and $[\cdot, \cdot]$ means concatenation for vectors. CFLP adopts the binary cross entropy loss to supervise the training process, which can be written as:

$$\mathcal{L}_F = \frac{1}{N^2} \sum_{i=1}^{N} \sum_{j=1}^{N} \ell(\mathbf{A}_{ij}, \widehat{\mathbf{A}}_{ij}) \\ \mathcal{L}_{CF} = \frac{1}{N^2} \sum_{i=1}^{N} \sum_{j=1}^{N} \ell(\mathbf{A}_{ij}^{CF}, \widehat{\mathbf{A}}_{ij}^{CF}) \tag{50}$$

where $\ell(\mathbf{A}_{ij}, \widehat{\mathbf{A}}_{ij})$ denotes the cross entropy loss between $\mathbf{A}_{ij}$ and $\widehat{\mathbf{A}}_{ij}$, $\mathcal{L}_F$ is the loss for the factual link prediction loss and $\mathcal{L}_{CF}$ is the counterfactual link prediction loss. Let $\hat{P}_f^F$ and $\hat{P}_f^{CF}$ be the node pair representations learned by graph encoder $f$ from factual distribution and counterfactual distribution, i.e., $[\mathbf{z}_i \odot \mathbf{z}_j, \mathbf{T}_{ij}]$ and $\left[\mathbf{z}_i \odot \mathbf{z}_j, \mathbf{T}_{ij}^{CF}\right]$. Since the test data contains only factual links at the inference stage, the model may suffer from the risk of covariant shift. Thus, CFLP adopts the discrepancy distance to regularize the representation learning, i.e., minimize the discrepancy of learned factual representation distributions $\hat{P}_f^F$ and learned counterfactual representation distribution $\hat{P}_f^{CF}$:

$$\mathcal{L}_{\text{disc}} = \text{disc}(\hat{P}_f^F, \hat{P}_f^{CF}) \tag{51}$$

where $\text{disc}(P, Q) = \|P - Q\|_F$. Combining the above link prediction loss and the discrepancy-minimization regularization, the overall training loss of CFLP is $\mathcal{L} = \mathcal{L}_F + \alpha \times \mathcal{L}_{CF} + \beta \times \mathcal{L}_{disc}$, where $\alpha$ and $\beta$ are hyperparameters to control the weights of counterfactual link prediction loss and discrepancy-minimization loss.

## 5.2 Counterfactual knowledge graph completion

Counterfactual knowledge graph completion is a natural extension of counterfactual link prediction. Knowledge graphs are heterogeneous graphs which have multiple types of nodes and edges[175]. For example, a knowledge graph may have entities such as persons, organizations and locations, and relations between them may include "is located in", "work for", or "has headquarters in". Knowledge graph completion (KGC) is an essential task on knowledge graphs, which aims to predict missing relationships in a knowledge graph based on the existing relationships between entities. Among existing KGC approaches, embedding-based KGC methods with GNNs[176, 177], which learn representations for entities in the embedding space with specifically designed GNNs, have become a popular choice. For example, R-GCN[178] uses relation-specific transformations and graph convolution to learn entity representations.

Since there are many different types of relations and the relation types are usually imbalanced on knowledge graphs, GNN-based KGC methods are found to have difficulty in properly modeling the imbalanced distribution of relations[163]. Motivated by the causal relationship among the entities on a knowledge graph, KGCF[163] aims to alleviate this issue by answering a counterfactual question: "would the relation still exist if the neighborhood of entities became different from observation?". Specifically, it generates counterfactual relations by treating the representations of entity pair given relation as context, structural information of relation-aware neighborhood as treatment, and validity of the composed triplet as the outcome. With the augmented knowledge graph, the downstream GNN-based model can learn more balanced relations and facilitate pair representation learning for KGC. KGCF extends the idea of CFLP[30]. It also uses a clustering method to obtain inter-community edges and intra-community edges as different treatments. The main difference between KGCF and CFLP is that the links can have different types. Thus, KGCF makes several changes on CFLP: 1) In (47), to find counterfactual links, CFLP does not consider the relation type; While KGCF requires that the counterfactual link should have the same relation type as the corresponding factual link; 2) To better model the relations in the representation learning step, KGCF utilizes NBFNet[179] as the encoder, which is an advanced graph learning framework for formulating different types of relations in knowledge graphs.

## 5.3 Counterfactual recommendation

Recommendation is a widely-used application of link prediction and a crucial service in information systems[5] and it can also be viewed as link prediction task. In this section, we first introduce the background of graph-based recommender systems[180–182] and the motivation for counterfactual recommendation. We then review recent advances in counterfactual recommendation.

### 5.3.1 Background and motivation

Recommender systems can rescue users from information overload and find the information they need in a timely manner. During the past decades, the mainstream

of recommender systems has evolved from neighborhood-based methods[183, 184] to representation learning-based methods[185, 186]. Neighborhood-based methods record historical-interacted items and recommend similar items to users[184]. In recent years, a variety of recommendation models have been proposed to learn user and item representations, ranging from matrix factorization techniques[187–189] to deep learning methods[186, 190]. Generally, user-item interactions can be modeled as a bipartite graph, with users and items as nodes and edges representing complex relationships between them. Leveraging these graphs, graph-based representation learning approaches[181, 191–193] have demonstrated promising performance in recommendation tasks. For instance, HiGNN[181] constructs new coarsened user-item graphs by clustering similar users or items and using the clustered centers as new nodes, enabling the model to capture hierarchical relationships among users and items more effectively.

Despite the success of existing graph-based recommender systems, they might make predictions with misleading spurious factors[194–196]. For example, some of the recommender systems prefer to recommend items with attractive exposure features. But when the actual content cannot match the exposure features, the recommendation will disappoint users. Here the exposure features work as the misleading spurious factor and we want to mitigate the influence of the spurious factor. Another example is the movie recommender system. Alice watched a lot of English movies, hence the recommender systems tend to think Alice enjoys English movies and continues to recommend English movies to her. However, the real reason for Alice watching those movies might be that she likes movies from the genre of Sci-Fi and many Sci-Fi movies are in English[160]. The language factor is the misleading spurious factor in this case. Many of the existing recommender systems are fooled by these misleading factors and fail to provide high-quality recommendations. Inspired by the causal theory, counterfactual recommendation attempts to build determined causal relationships among involved variables and remove the misleading spurious factors.

#### 5.3.2 Methodologies

There are several attempts on counterfactual recommendation to remove spurious factors in various scenarios[158–162, 164, 165]. Next, we will briefly introduce them.

**Counterfactual recommendation for mitigating clickbait issue.** Many recommender systems try to optimize the click-through-rate (CTR)[197, 198], i.e., maximizing the likelihood that a user clicks the recommended items. However, the clickbait issue widely exists in this scenario, which utilizes attractive exposure features $E$ to attract clicks but shows disappointing content $T$ after clicking[158]. Given the misleading exposure features, a click from user $u$ to item $i$ cannot reflect the true preference of $u$ to $i$. Hence, recommender systems trained on such datasets will tend to recommend items with attractive exposure features with potentially disappointing content, which significantly hurts user experiences. As shown in Fig. 6, the exposure features $E$ can influence click $Y$ in two ways: 1) natural direct effect (NDE) that causes clickbait issue, i.e., $E \to Y$; and 2) total indirect effect (TIE) as part of item features to influence the prediction score, i.e., $E \to I \to Y$. The total effect of exposure features $E$ on prediction score $Y$ is modeled as total effect (TE). The relationship of these three effects is $\text{TE} = \text{NDE} + \text{TIE}$. Based on this causal graph, Wang et al.[158] propose CR to alleviate the issue by removing the direct influence of exposure features, i.e., $\text{TIE} = \text{TE} - \text{NDE}$. The influence of item features on prediction score TIE is kept as the prediction score $\hat{Y}$ to provide a desired recommendation without clickbait issue.

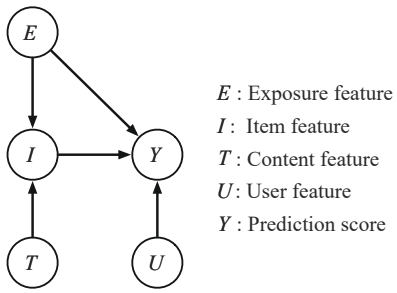



Fig. 6 Causal graph for clickbait issue[158]

Specifically, let $Y_{u,i} \in \{0, 1\}$ be whether user $u$ clicks on item $i$ $(Y_{u,i} = 1)$ or not $(Y_{u,i} = 0)$. The predicted clicking probability of user $u$ on item $i$ is $\hat{Y}_{u,i} = s_\theta(u, i)$, with $s_\theta(\cdot, \cdot)$ being a scoring function parameterized by $\theta$. The recommendation task involves learning an effective scoring function on the training set by minimizing the loss:

$$\theta = \arg\min_{\theta} \sum_{(u,i,Y_{u,i}) \in \mathcal{D}} l\left(s_\theta(u, i), Y_{u,i}\right) \tag{52}$$

where $l(\cdot, \cdot)$ is the cross-entropy loss and $\mathcal{D}$ denotes the training set. In the clickbait context, item features comprise exposure features $e$ and content features $t$, i.e., $i = (e, t)$. Both item and exposure features affect the prediction score, as $\hat{Y}_{u,i,e} = f(u, i, e)$. To model the direct effect of one variable on another while keeping other variables fixed, we should maintain $I$ constant and prevent it from contributing to $Y$. The authors use reference value $I = i^*$ to represent the absence of $I$ information, i.e., $Y_{u,i^*,e} := Y_{u,e}$, to signify that the variable $I$ lacks distinctive characteristics. Consequently, the natural direct effect of $I$ on $Y$ can be expressed as $\text{NDE} = Y_{u,i^*,e} - Y_{u,i^*,e^*}$, in the absence of $i^*, e^*$. Then, the prediction score can be obtained as $\hat{Y}_{\text{CR}} = \text{TE} - \text{NDE}$. CR estimates NDE and TE as:

$$\text{NDE} = \hat{Y}_{u,i^*,e} - \hat{Y}_{u,i^*,e^*}, \text{TE} = \hat{Y}_{u,i,e} - \hat{Y}_{u,i^*,e^*}. \tag{53}$$

Thus, CR acquires the desired prediction by subtracting NDE from TE, i.e.,

$$\hat{Y}_{\mathrm{CR}} = \mathrm{TE} - \mathrm{NDE} = \hat{Y}_{u,i,e} - \hat{Y}_{u,i^*,e}. \tag{54}$$

Since user-item interactions are modeled as a graph, CR employs a GNN model called multi-modal graph convolution network (MMGCN)[199] as the base model. Both $\hat{Y}_{u,i} = f_{\theta_1}(U = u, I = i)$ and $\hat{Y}_{u,e} = f_{\theta_2}(U = u, E = e)$ are modeled separately, and the fusion strategy is applied as $\hat{Y}_{u,i,e} = \hat{Y}_{u,i} \times \sigma(\hat{Y}_{u,e})$. To optimize the model, both $\hat{Y}_{u,i,e}$ and $\hat{Y}_{u,i^*,e}$ need improvement. Noting that $\hat{Y}_{u,i^*,e} = \hat{Y}_{u,e}$, considering we only have $Y_{u,i}$ as the supervision, the objective function can be expressed as:

$$\min_{\theta} \sum_{(u,i,Y_{u,i}) \in \mathcal{D}} l(\hat{Y}_{u,i,e}, Y_{u,i}) + \alpha \times l(\hat{Y}_{u,e}, Y_{u,i})$$

where $\alpha$ is a hyper-parameter. With this design, both $\hat{Y}_{u,i,e}$ and $\hat{Y}_{u,i^*,e}$ can be estimated, enabling the acquisition of an accurate prediction, $\hat{Y}_{\mathrm{CR}}$.

**Counterfactual recommendation for mitigating geographical factor.** Point-of-interests (POIs) recommendations are widely used in location-based services[200–202]. In many existing location recommendation systems, the interactions between users and POIs are determined by the interests of users and the functional attributes of POIs. Recently, Liu et al.[161] notice that the geographical factor works as a confounder of this interaction, i.e., the geographical factor not only affects the POIs, but also influences the user-POI interactions. For example, in downtown, there are many restaurants but Alice prefers to go to restaurant $r$. The reasons may differ: Alice may be attracted by the restaurant, or maybe Alice chooses this restaurant just because it is near her office (geographical factor). Taking this into consideration, Liu et al.[161] propose UKGC, which can remove the direct influence of geographical factor on user-POI interactions with counterfactual learning and make recommendations merely based on user features and POI features. The core idea of UKGC is to model the geographical factor separately with disentangled representation learning, and then use counterfactual learning to remove the direct effect of geographical bias. Specifically, in the first step, UKGC collects various POI information and cities' geographical data from Tencent map and builds a large-scale urban knowledge graph (UrbanKG). It then divides UrbanKG into two subgraphs, the geographical graph and the functional graph, with respect to the type of relations. Both graphs contain the same POIs. UKGC adopts graph convolutions to learn geographical embeddings and functional embeddings for users and POIs then integrates two sets of embeddings via linear combination and gets the final embedding of users and POIs. In the second step, UKGC shares the same design of CR[158] to mitigate geographical factors with counterfactual learning. Formally, let $Y_{u,p,g}$ denote the recommendation score between use $u$ and POI $p$, where $g$ is the geographical attribute of $p$. Similar to CR, the natural direct effect (NDE) of the geographical factor on recommendation is:

$$\mathrm{NDE} = Y_{u,p^*,g} - Y_{u,p^*,g^*} \tag{55}$$

where $p^*$ and $g^*$ are the reference value of $p$ and $g$. The reference value signifies that the variable lacks distinctive characteristics, and it is identical for each variable, computed as the average value. Similarly, the total effect (TE) and total indirect effect (TIE) of geographical factors on recommendation are defined as:

$$\begin{aligned} \mathrm{TE} &= Y_{u,p,g} - Y_{u,p^*,g^*} \\ \mathrm{TIE} &= \mathrm{TE} - \mathrm{NDE} = Y_{u,p,g} - Y_{u,p^*,g} \end{aligned} \tag{56}$$

where TIE is the desired recommendation score between user $u$ and POI $p$ without the direct influence of geographical factors. The model optimization closely resembles the CR approach[158]. Interested readers can refer to the original paper[161] for more details.

**Counterfactual recommendation for alleviating spurious correlation.** Instead of mitigating a single predetermined spurious factor in recommender systems, CGKR[160] aims to integrate information from knowledge graphs (KGs) to simultaneously identify and alleviate potential spurious correlations. For instance, John has watched numerous action movies featuring car chases, causing the recommender system to suggest more movies with car chases for him. However, the actual reason behind John's preference for these action movies is his admiration for a particular actor who often stars in films with car chase scenes. Thus, the co-occurrence patterns may mislead the recommender to make recommendations based on the spurious factor, i.e., the car chase factor in this case. CGKR addresses this problem by designing counterfactual generators. The mutual collaboration mechanism between the recommender and the generators helps to find potential spurious correlations and weaken their influence. The whole process can be illustrated using John's example. Intuitively, knowing that "John watched a movie" and the movie features a car chase from the training data, CGKR generates a counterfactual interaction by asking and answering the question, "would John still watch the movie if it didn't have a car chase scene?". If the recommender answers yes, the spurious correlation of car chases will be identified and removed.

In the first step, CGKR specifically designs counterfactual generators to modify item attributes, enabling the creation of desired counterfactual interactions. Given the vast space of potential modifications, CGKR employs a Markov decision process[203] (MDP) to streamline the process. To guide the MDP in discovering spurious counterfactual interactions, two rewards are designed: the in-

formation-based reward, which helps identify essential attributes, and the prediction-based reward, which facilitates generating more informative counterfactual interactions. For the second step, CGKR employs a knowledge-aware recommender that jointly leverages knowledge graphs and user-item interactions to learn item and user representations. It utilizes an inductive graph neural network (GNN) to learn both factual and counterfactual item representations from the knowledge graph. Subsequently, a GNN is employed to aggregate user-item interactions and obtain user representations. The recommender is supervised to provide similar predictions for both factual and counterfactual interactions, ensuring reliable recommendations without spurious interactions.

**Counterfactual recommendation for using outer-session causes.** Different from previous session-based recommendation methods that focus on the inner-session causes, COCO-SBRS[162] proves the effectiveness of considering both outer factors and inner-session causes. It uses counterfactuals and collaborative filtering to alleviate the spurious correlation between inner-session causes and prediction. COCO-SBRS consists of three stages: abduction, action, and prediction[162]. In the abduction stage, they design and create a base recommendation model with self-supervised learning techniques. This model considers both inner-session causes and outer-session causes based on their designed causal graphs. The action stage and prediction stage make sure the next item is created for similar session only when users′ inner-session causes and outer causes match those of the target session. COCO-SBRS utilizes the designed base recommendation model in the first stage to simulate human decision given counterfactual questions.

**Counterfactual recommendation for mitigating bias from User-Item interaction.** FairGap[164] aims to create an unbiased counterfactual graph based on observed interactions to achieve fair recommendations. FairGap first constructs a structural bipartite graph where nodes have no attributes. It then generates biased (observed) bipartite graphs and unbiased (counterfactual) bipartite graphs based on the structural bipartite graph. Because these two kinds of graphs are both generated from the structural bipartite graph, FairGap designs the optimization objective by comparing the similarity in the patterns of connections between them. To further refined the learned embeddings, FairGap uses Jenson-Shannon divergence to create another loss function that only considers user groups that share the same attributes.

**Counterfactual recommendation for mitigating consumer unfairness.** Boratto et al.[165] utilize counterfactual explanation methods to generate new user-item interactions. The new interactions are designed to enhance fairness based on the explanations. Boratto et al.[165] design a loss function containing two components in which one focuses on ensuring demographic parity among user groups while another one controls the extent of graph modification.

**Counterfactual explanation in recommender systems.** GREASE[159] is a natural extension of GNN counterfactual explanation method CF$^2$[159] for recommender systems. To mitigate the gap between recommendation and explanation, Chen et al.[159] propose GREASE to find the sufficient and necessary conditions for an item to be recommended, respectively. The proposed GREASE has two steps, i.e., training a surrogate model for the target model and generating explanations with adjacency matrix perturbation. For the first step, GREASE adopts a relational graph convolutional network (R-GCN)[4] as the surrogate model by minimizing the mean squared error of the node embeddings of the target model and that of the surrogate model. For the second step, GREASE defines two adjacency matrix masks for factual explanation and counterfactual explanation, separately. Considering the case that recommends item $i$ to user $u$, the mask for factual explanation is used to perturb the adjacency matrix and seek to maximize the recommendation score for $i$ until $i$ is recommended. Similarly, Chen et al.[159] perturb the adjacency matrix with a counterfactual mask and try to minimize the recommendation score until $i$ is no longer recommended. These learned factual and counterfactual masks can be used to indicate the importance of user-item interactions in the adjacency matrix and explain the model.

## 6 Real-world applications of graph counterfactual learning

Due to its superior performance in many tasks[47, 92], graph counterfactual learning has wide applications in real-world scenarios. In this section, we will first review applications in various domains, including physical systems, medical and molecular, which is summarized in Table 7. We will then review the adoption of graph counterfactual learning to facilitate other machine learning techniques, which is summarized in Table 8.

### 6.1 Applications of graph counterfactual learning

In this subsection, we review applications of graph counterfactual learning in various domains.

#### 6.1.1 Physical systems

The ability to find the underneath causal relations is a representative feature of intelligent agents. As a human, we have the capacity to discover causal effects in the real-world physical environment. For example, having observed a ball being thrown in a certain manner, we can predict its parabolic trajectory. Counterfactual learning together with graph neural networks makes it possible to model the underlying causal structure of complex systems, enabling intelligent agents to make more accurate predictions and reason about potential interven-

Table 7 Summary of real-world applications of graph counterfactual learning

| Category | Methods | Graph types | Tasks |
|---|---|---|---|
| Physical systems | CoPhy[204], V-CDN[205], Li et al.[206] | Objects with interactions | Position prediction |
| | XHGP[207] | Vehicles, pedestrians with interactions | Position prediction |
| | TGV-CRN[208], Su et al[209] | Multi-agents and interactions | Position prediction |
| | Naderializadeh et al.[210] | Wireless interference network | Power allocation |
| Medical | CMGE[211] | EMR text graph | Explain diagnosis result |
| | CACHE[212] | Hypergraph of EMR | Explain clinical predictions |
| | Zhang et al.[213] | Medical images and their relations | Disease prediction |
| Molecular | SolvGNN[214] | Molecule graph | Molecule property prediction |
| | MMACE[215] | Molecule graph | Explain property prediction |

Table 8 Graph counterfactual learning for other techniques

| Techniques | References |
|---|---|
| Transfer learning | Li et al.[216] |
| Data augmentation | CRL[217] |
| Anomaly detection | CFAD[218], CFGAD[219] |

tions[204–206].

In physical systems, predicting alternative future trajectories for objects within a 3D simulation is an important problem[204]. Specifically, with a video depicting the motion of objects in one scenario, we aim to predict the trajectory of the object given a single altered frame that modifies object positions at the initial timestep of the input video. The task can be an effective way to evaluate the model performance in terms of counterfactual reasoning and their ability to model complex systems. CoPhy[204] and V-CDN[205] adopt end-to-end approaches to learn latent representations of confounders and perform counterfactual reasoning, allowing for more accurate predictions of alternative future trajectories when compared to feedforward video prediction baselines.

Specifically, these two models build graph structure from the video to model the complex interactions between objects and perform interventions on the graph to get counterfactual outcomes. This enables precise predictions and possible interventions in the real-world physical environment, such as predicting alternative future trajectories of objects in the aforementioned task.

Though they are effective in modeling causal relationships and counterfactual reasoning, CoPhy and V-CDN are prone to overfitting. To alleviate the overfitting issue, Li et al.[206] take the unobservable confounders into consideration, such as masses and gravity that are tied to physics laws and independent of the data. By incorporating confounders into the model, Li et al.[206] enhance the model's ability to capture the true underlying causal structure, reducing the risk of overfitting.

In addition to the ability to perform counterfactual reasoning, another advantage of counterfactual learning is to help investigate the model behavior in different scenarios. The intuition is that properly modeling the causal relationships of variables in a system can help to identify the critical variables that strongly affect the outcomes[42]. One example is interpretable motion prediction models, which can enable autonomous vehicles to perform safe and transparent decisions[220]. Specifically, the motion prediction system is to estimate future trajectories of the vehicle of interest with input features describing the past trajectory of the vehicle as well as the scene context. Limeros et al.[207] propose XHGP, which uses a graph-based model to model heterogeneous objects in traffic scenes and interactions between objects. It adopts counterfactual learning to examine the trained models' sensitivity to changes made in the input data, such as masking scene elements, modifying trajectories, and adding or removing dynamic agents.

Graph counterfactual learning has also been applied to the domain of multi-agent systems. Multi-agent systems involve the interaction and cooperation of multiple autonomous agents, each with their own goals, to achieve a common objective or solve a problem[221]. For example, in the network packet routing problem, multiple autonomous routers work together to forward packets through a network efficiently and quickly[221]. In multi-agent systems, partial observability and communication constraints cause two crucial problems: How to make efficient communications between objects and how to assign credit from a global reward function. Su et al.[209] tackle the first problem by modeling agents and interactions with graphs and utilizing graph convolution to integrate desired information from neighboring nodes. They settle the second problem by incorporating COMA[222], which makes counterfactual reasoning for each agent to understand the sole contribution and assign credit accordingly. Fujii et al.[208] propose TGV-CRN to estimate the effect of intervention in multi-agent systems in interpretable ways. They model confounders with graph representation learning techniques and make predictions with the intervened representations to estimate the effect. The result can help researchers understand the potential effect of an intervention, e.g., in autonomous vehicle simulation, what

is the potential effect of the acceleration of a vehicle. Wireless power control problem has multiple transmitter-receiver pairs communicating with each other, which can also be treated as a multi-agent system. Naderializadeh et al.[210] use a GNN-based framework to learn optimal power allocation decisions with the help of counterfactuals.

#### 6.1.2 Medical

In medical domain, it is critical to know the supporting facts of the clinical predictions to ensure the transparency and reliability of the diagnoses and treatments[211]. By employing graph counterfactual learning in medical research, it becomes possible to identify the underlying causal relationships between different variables, such as symptoms, diagnoses and treatments, allowing for more accurate predictions and evidence-based decision-making[211–213].

One example is explainable electronic medical records (EMR) diagnosis, which aims to make an explainable clinical diagnosis with machine learning models[223]. The common approach is to treat the EMR as a text sequence and utilize an external expert knowledge base to find explanations for clinical diagnosis. CMGE[211] leverages counterfactual learning to gradually weaken the features until the diagnosis changes dramatically, thus the feature could be considered as the supporting fact. It adopts graph modeling techniques to provide multi-granularity entities, such as sentences, clauses and words, together with the relationships at different levels. Thus, CMGE can avoid external knowledge and try to provide explanations appropriately for the diagnosis solely with the EMR itself. Facing the same problem of providing explanations for EMR-based clinical diagnosis, CACHE[212] models the relationships between medical entities with hypergraphs. Hypergraphs are generalizations of traditional graph structure that allow edges to connect more than two nodes, which are called hyperedges. In CMGE, hyperedges connect multiple entities, such as chest pain, obesity and gout together, to represent the complex relations among these units. Counterfactual learning is also used to find features that cause the change of predictions as the critical supporting fact. Zhang et al.[213] endeavor to improve disease prediction performance by learning the cause-effect relationship from data. They ask counterfactual questions "would someone be inferred to be free of the disease if he/she has some attributes?". By answering the counterfactual questions, their model can learn the latent relationships and integrate knowledge into their prediction model.

Holzinger et al.[224] highlight the importance of incorporating various modalities in medical domain. They propose using GNNs to construct multi-modal representations, spanning images, text and genomics data, where each sample works as a node and they are connected based on various relationships. The flexibility of GNNs can effectively enable information fusion for multiple models. In this process, counterfactual learning works as an interpretable tool to investigate the model behavior in various scenarios by exploring how changes made to the input data affect the outcomes.

#### 6.1.3 Molecular

In molecular studies, it is important to understand the impact of functional groups and composition on equilibrium behavior[8]. A deep understanding of this problem can greatly help in designing new materials, optimizing chemical processes, and predicting the properties of complex systems. GNNs have gained wide popularity in molecular studies due to their ability to capture and represent intricate relationships between different molecular components in multi-component systems[225]. However, the lack of transparency severely limits the border application of GNNs in molecular studies[226]. SolvGNN[214] tackles this problem by incorporating counterfactual learning. It performs counterfactual analysis by perturbing the input graph structure until the prediction changes. The counterfactual analysis helps gain a further understanding of which chemical structures and functional groups can lead to certain activity coefficient predictions. Continuing this line of research, MMACE[215] is proposed to generate experimentally available molecules as counterfactuals by enumerating chemical space. Thus, MMACE is independent of the target model architecture and can be applied to explain various models.

### 6.2 Graph counterfactual learning for other techniques

#### 6.2.1 Transfer learning

Transfer learning aims to transfer knowledge from source domains to target domains to facilitate the task in the target domain. Li et al.[216] focus on a specific setting, the relational transfer learning task, which extracts semantic-meaningful entities to build knowledge graphs in the source domain and then transfer the structured knowledge to the target domain. However, a critical challenge of relational transfer learning is that it needs support from human experts to select appropriate relations to transfer, which is time-consuming and laborious. Li et al.[216] tackle this problem by employing counterfactual inference. With counterfactual inference, one can discover latent causal relationships, which can help predict causal knowledge graphs from the source domain to the target domain for relational transfer learning.

#### 6.2.2 Data augmentation

In previous sections, we have shown the effectiveness of performing counterfactual augmentation, such as counterfactual link prediction[30]. The intuition is that the counterfactual augmented data can help the model learn the latent causal relationships better. For example, in the image classification task, an image with a bird on it is labeled as a "bird" class. Counterfactual augmentation aims to remove the bird in the image and label the aug-

mented image as "non-bird" class. By doing so, the model learns to identify the bird as the key causal feature associated with the "bird" class, thereby improving its understanding of the causal relationships between image features and class labels. CRL[217] employs counterfactual learning in a reinforcement learning-based manner for graph classification task. It integrates external domain knowledge to guide the counterfactual data augmentation and then uses GNNs to learn from both the original and counterfactual augmented data.

#### 6.2.3 Anomaly detection

Anomaly detection in attributed networks aims to identify unusual or suspicious nodes within a network that contains both topological structure and node attribute. It is widely adopted in various domains such as social networks and transaction networks[227]. However, the distribution discrepancy of training data and test data may hurt the generalization ability of existing anomaly detection approaches since they are trained by learning environment-dependent correlations. That means training nodes and test nodes are different with respect to their environments, e.g., subgraph topology and neighboring nodes. To solve this problem, CFAD[218] constructs the causal graph to learn the causal relations among latent and observed variables. Specifically, CFAD constructs the counterfactual examples in different environments to help the model generalize to unseen scenarios. With the root causes captured and spurious information filtered, CFAD can make robust and stable predictions in anomaly detection. CFGAD[219] uses a weight-varying diffusion model to create counterfactual samples using class and environment features separated by a gradient-based feature selector. GNNs can be more robust after being trained on these samples.

## 7 Future directions

Despite various efforts taken on graph counterfactual learning, it is still in its early stage. Existing approaches mainly focus on static graphs and lack the ability to generalize to other settings, such as large-scale graphs and dynamic graphs. Moreover, most existing datasets and metrics are developed for factual learning, not dedicated to counterfactual learning, which may not accurately reflect the performance of counterfactual learning methods. Additionally, some directions are not yet fully explored and require further investigation, such as unsupervised graph counterfactual learning and counterfactual data augmentation. We summarize the existing open-source implementations in Table 9 to help readers have a better knowledge of current progress. Next, we give the details of future works, aiming to provide researchers with insights into crucial directions in this field.

**1) Benchmark datasets.** A well-curated dataset can significantly accelerate the development of a research area[229]. Currently, graph counterfactual learning lacks benchmark datasets. For real-world datasets, we typically have access only to observed data, making it challenging to evaluate the performance of counterfactual learning methods. Thus, one important future research direction is constructing synthetic or semi-synthetic datasets that encompass various characteristics of real-world graphs while providing ground truth for counterfactual scenarios[42]. For example, in counterfactual fairness, researchers may come up with synthetic social networks where each node has node attributes and connections in the counterfactual world where the node sensitive attribute is flipped. For semi-synthetic datasets, in computer vision, researchers use deepfake to change the face in an image to a different gender[230], such that they can get the counterfactuals for fair face recognition tasks. For graph domains, we need to design specific approaches to generate counterfactual nodes or edges within the context of a specific application or domain. There is also a lack of benchmarks for validating the accuracy of the spurious factors identified. One often needs domain experts or domain knowledge in order to validate the spurious factors identified, which is time-consuming and costly. In addition, the datasets should be diverse and cover a wide range of applications, such as social networks, biological systems, and financial markets, to enable a comprehensive evaluation of different graph counterfactual learning techniques.

**2) Evaluation metrics.** Another important problem is that graph counterfactual learning lacks dedicated evaluation metrics. Currently, existing graph counterfactual learning methods adopt evaluation metrics designed for conventional graph learning. For example, statistical parity is a group fairness metric and most of graph counterfactual learning works also adopt this metric for counterfactual fairness. Although it can effectively compare the prediction result of different sub-populations, it has been found not able to detect discrimination in presence of statistical anomalies[48]. For graph counterfactual explanation, it also faces the same problem that existing metrics may not be well-suited for evaluating the quality of counterfactual explanations generated for graph-structured data. To address these issues, the development of tailored metrics to facilitate evaluation of counterfactual learning on graphs is a promising and urgent direction. These metrics should align well with the causal structure of the problem formulation.

**3) Counterfactual data augmentation.** Counterfactual learning typically involves: i) modeling counterfactual outcomes[30, 217]; ii) identifying root causes of model predictions[7, 159]. Counterfactual augmentation focuses on modeling outcomes, providing additional knowledge to enhance input-prediction relationships. For example, in animal image classification, counterfactual augmentation removes a bird, labeling the altered image as "non-bird." In graph domains, such as recommendations based on user connections, augmentation may involve removing a

Table 9 Summary of open-source implementations

| Methods | Categories | Framework | Code |
|---|---|---|---|
| GEAR[9] | Fairness | PyTorch | https://github.com/jma712/GEAR |
| NIFTY[92] | Fairness | PyTorch | https://github.com/chirag126/nifty |
| CAF[106] | Fairness | PyTorch | https://github.com/TimeLovercc/CAF-GNN |
| GraphXAI[228] | Explanation | PyTorch | https://github.com/mims-harvard/GraphXAI |
| GCFExplainer[134] | Explanation | PyTorch | https://github.com/mertkosan/GCFExplainer |
| Wellawatte et al.[215] | Explanation | PyTorch | https://github.com/ur-whitelab/exmol |
| GRETEL[146] | Explanation | TensorFlow | https://github.com/MarioTheOne/GRETEL |
| $CF^2$ [35] | Explanation | PyTorch | https://github.com/chrisjtan/gnn_cff |
| CF-GNNExplainer[47] | Explanation | PyTorch | https://github.com/a-lucic/cf-gnnexplainer |
| RCExplainer[129] | Explanation | PyTorch | https://github.com/RomanOort/FACTAI |
| MEG[7] | Explanation | PyTorch | https://github.com/danilonumeroso/meg |
| Abrate and Bonchi[132] | Explanation | PyTorch | https://github.com/carlo-abrate/CounterfactualGraphs |
| Chhablani et al.[147] | Explanation | PyTorch | https://github.com/sarthakja/Game-Theoretic-Counterfactual-Explanation-GNN |
| CFLP[30] | Link prediction | PyTorch | https://github.com/DM2-ND/CFLP |
| CR[158] | Recommendation | PyTorch | https://github.com/WenjieWWJ/Clickbait/ |
| CGKR[160] | Recommendation | PyTorch | https://github.com/RUCAIBox/CGKR |
| COCO-SBRS[162] | Recommendation | PyTorch | https://github.com/wzsong17/COCO-SBRS |
| Boratto et al.[165] | Recommendation | PyTorch | https://github.com/jackmedda/RS-BGExplainer/tree/cikm2023 |
| XHGP[207] | Physical system | PyTorch | https://github.com/sancarlim/Explainable-MP/tree/v1.1 |
| TGV-CRN[208] | Physical system | PyTorch | https://github.com/TGV-CRN/TGV-CRN |
| V-CDN[205] | Physical system | PyTorch | https://github.com/pairlab/v-cdn |
| CoPhy[204] | Physical system | PyTorch | https://projet.liris.cnrs.fr/cophy/ |
| SolvGNN[214] | Molecular | PyTorch | https://github.com/zavalab/ML/tree/SolvGNN |
| CMGE[211] | Medical | PyTorch | https://github.com/CKRE/CMGE/tree/main |

user's influential connection and labeling altered graph as a negative example. This highlights certain connections and improves causal understanding between connections and labels. Despite limited initial efforts in graph domains[30], ample opportunities exist for further research, including tailored strategies for specific applications like recommendation systems[62, 190, 196, 231], social network analysis[64, 232], or molecular graph models[25].

**4) Scalability.** Large-scale graphs are pervasive in real-world such as Facebook or Twitter[6, 233], web graphs[234] and knowledge graph[235, 236]. Due to the complex graph structure, the iterative process of generating counterfactuals and the need to retrain models with augmented data, existing graph counterfactual learning approaches are time-consuming and often struggle with scalability, making them unsuitable for large-scale graphs[237]. Despite the importance of scalability, works on addressing this issue in graph counterfactual learning are relatively limited.

**5) Dynamicity.** Many real-world graphs are inherently dynamic[238], such as social networks where new relationships form and old ones dissolve over time[239]; transportation networks where traffic patterns and infrastructure change[240] and user-item interaction networks in recommender systems where relationships between users and items evolve[241, 242]. However, existing graph counterfactual learning approaches mainly focus on static graphs. To deal with the dynamic graphs effectively, we need to address several challenges, such as the complexity of modeling the graph structure over time and the lack of high-quality dynamic graph datasets.

**6) Unsupervised learning.** Unsupervised/self-supervised learning on graphs have achieved comparable performance with supervised methods[232, 243, 244]. However, there are very few explorations for unsupervised/self-supervised graph counterfactual learning. The lack of labeled data in unsupervised setting makes it difficult to investigate the causal structure of latent variables and observed variables. Moreover, it is challenging to generate counterfactual outcomes and compare them with factual outcomes without the necessary ground truth label[245]. Given the success of unsupervised learning in graph domain, it is crucial to address this gap and explore innovative approaches for unsupervised graph counterfactual learning.

## 8 Conclusions

In this survey, we present a comprehensive review of counterfactual learning on graphs from the problems of counterfactual fairness, counterfactual explanation, counterfactual link prediction and the real-world applications of graph counterfactual learning. This is the first survey for counterfactual learning on graphs. In particular, we first introduce the basic concept of counterfactual learning, then introduce a framework to give a unified understanding of the problems. We also summarise the datasets and metrics used in each category. Then we go beyond the counterfactual learning on graphs to its applications in many areas, such as physical systems, medical, etc. Finally, we also discuss future directions and encourage domain experts to contribute to essential and urgent topics in this area. We believe this survey can give starters the fundamental knowledge and inspire the domain experts to solve the urgent challenges in this area.

## Acknowledgements

This material was based upon work supported by, or in part by the National Science Foundation (NSF), USA (No. IIS-1909702), Army Research Office (ARO), USA (No. W911NF-21-10198), and Cisco Faculty Research Award.

## Declarations of conflict of interest

The authors declared that they have no conflicts of interest to this work.

## Open Access



## References

[1] W. L. Hamilton. *Graph Representation Learning*, Cham, Switzerland: Springer, 2020. DOI: 10.1007/978-3-031-01588-5.

[2] S. Tabassum, F. S. F. Pereira, S. Fernandes, J. Gama. Social network analysis: An overview. *WIREs Data Mining and Knowledge Discovery*, vol. 8, no. 5, Article number e1256, 2018. DOI: 10.1002/widm.1256.

[3] C. W. Coley, W. G. Jin, L. Rogers, T. F. Jamison, T. S. Jaakkola, W. H. Green, R. Barzilay, K. F. Jensen. A graph-convolutional neural network model for the prediction of chemical reactivity. *Chemical Science*, vol. 10, no. 2, pp. 370–377, 2019. DOI: 10.1039/C8SC04228D.

[4] Z. Zhang, F. Z. Zhuang, H. S. Zhu, Z. P. Shi, H. Xiong, Q. He. Relational graph neural network with hierarchical attention for knowledge graph completion. In *Proceedings of the 34th AAAI Conference on Artificial Intelligence*, New York, USA, pp. 9612–9619, 2020. DOI: 10.1609/aaai.v34i05.6508.

[5] S. W. Wu, F. Sun, W. T. Zhang, X. Xie, B. Cui. Graph neural networks in recommender systems: A survey. *ACM Computing Surveys*, vol. 55, no. 5, Article number 97, 2023. DOI: 10.1145/3535101.

[6] A. L. Traud, P. J. Mucha, M. A. Porter. Social structure of facebook networks. *Physica A: Statistical Mechanics and its Applications*, vol. 391, no. 16, pp. 4165–4180, 2012. DOI: 10.1016/j.physa.2011.12.021.

[7] D. Numeroso, D. Bacciu. MEG: Generating molecular counterfactual explanations for deep graph networks. In *Proceedings of International Joint Conference on Neural Networks*, Shenzhen, China, 2021. DOI: 10.1109/IJCNN52387.2021.9534266.

[8] Y. Y. Wang, J. R. Wang, Z. L. Cao, A. B. Farimani. Molecular contrastive learning of representations via graph neural networks. *Nature Machine Intelligence*, vol. 4, no. 3, pp. 279–287, 2022. DOI: 10.1038/s42256-022-00447-x.

[9] Z. H. Wu, S. R. Pan, F. W. Chen, G. D. Long, C. Q. Zhang, P. S. Yu. A comprehensive survey on graph neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, no. 1, pp. 4–24, 2021. DOI: 10.1109/TNNLS.2020.2978386.

[10] J. Tang, M. Qu, M. Z. Wang, M. Zhang, J. Yan, Q. Z. Mei. LINE: Large-scale information network embedding. In *Proceedings of the 24th International Conference on World Wide Web*, Florence, Italy, pp. 1067–1077, 2015. DOI: 10.1145/2736277.2741093.

[11] A. Grover, J. Leskovec. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, San Francisco, USA, pp. 855–864, 2016. DOI: 10.1145/2939672.2939754.

[12] T. N. Kipf, M. Welling. Semi-supervised classification with graph convolutional networks. In *Proceedings of the 5th International Conference on Learning Representations*, Toulon, France, 2017.

[13] M. H. Zhang, Y. X. Chen. Link prediction based on graph neural networks. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, Montreal, Canada, pp. 5171–5181, 2018.

[14] O. Shchur, S. Günnemann. Overlapping community detection with graph neural networks, [Online], Available: https://arxiv.org/abs/1909.12201, 2019.

[15] Y. D. Sui, X. Wang, J. C. Wu, M. Lin, X. N. He, T. S. Chua. Causal attention for interpretable and generaliz-

able graph classification. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, Washington DC, USA, pp. 1696–1705, 2022. DOI: 10.1145/3534678.3539366.

[16] K. M. He, X. Y. Zhang, S. Q. Ren, J. Sun. Deep residual learning for image recognition. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, Las Vegas, USA, pp. 770–778, 2016. DOI: 10.1109/CVPR.2016.90.

[17] J. Devlin, M. W. Chang, K. Lee, K. Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Minneapolis, USA, pp. 4171–4186, 2019. DOI: 10.18653/V1/N19-1423.

[18] Z. Y. Liu, J. Zhou. Graph attention networks. *Introduction to Graph Neural Networks*, Z. Y. Liu, J. Zhou, Eds., Cham, Switzerland: Springer, pp. 39–41, 2020. DOI: 10.1007/978-3-031-01587-8_7.

[19] K. Y. L. Xu, W. H. Hu, J. Leskovec, S. Jegelka. How powerful are graph neural networks? In *Proceedings of the 7th International Conference on Learning Representations*, New Orleans, USA, 2019.

[20] M. Chen, Z. W. Wei, Z. F. Huang, B. L. Ding, Y. L. Li. Simple and deep graph convolutional networks. In *Proceedings of the 37th International Conference on Machine Learning*, pp. 1725–1735, 2020.

[21] J. Gasteiger, A. Bojchevski, S. Günnemann. Predict then propagate: Graph neural networks meet personalized pagerank. In *Proceedings of the 7th International Conference on Learning Representations*, New Orleans, USA, 2019.

[22] D. Kim, A. Oh. How to find your friendly neighborhood: Graph attention design with self-supervision. In *Proceedings of the 9th International Conference on Learning Representations*, 2021.

[23] L. M. Pan, C. Shi, I. Dokmanić. Neural link prediction with walk pooling. In *Proceedings of the 10th International Conference on Learning Representations*, 2022.

[24] P. A. Papp, K. Martinkus, L. Faber, R. Wattenhofer. DropGNN: Random dropouts increase the expressiveness of graph neural networks. In *Proceedings of the 35th International Conference on Neural Information Processing Systems*, 2021.

[25] Z. P. Xiong, D. Y. Wang, X. H. Liu, F. S. Zhong, X. Z. Wan, X. T. Li, Z. J. Li, X. M. Luo, K. X. Chen, H. L. Jiang, M. Y. Zheng. Pushing the boundaries of molecular representation for drug discovery with the graph attention mechanism. *Journal of Medicinal Chemistry*, vol. 63, no. 16, pp. 8749–8760, 2020. DOI: 10.1021/acs.jmedchem.9b00959.

[26] K. Shu, A. Sliva, S. H. Wang, J. L. Tang, H. Liu. Fake news detection on social media: A data mining perspective. *ACM SIGKDD Explorations Newsletter*, vol. 19, no. 1, pp. 22–36, 2017. DOI: 10.1145/3137597.3137600.

[27] C. H. Wu, F. Z. Wu, Y. Cao, Y. F. Huang, X. Xie. FedGNN: Federated graph neural network for privacy-preserving recommendation, [Online], Available: https://arxiv.org/abs/2102.04925, 2021.

[28] E. Y. Dai, S. H. Wang. Say no to the discrimination: Learning fair graph neural networks with limited sensitive attribute information. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*, pp. 680–688, 2021. DOI: 10.1145/3437963.3441752.

[29] E. Y. Dai, S. H. Wang. Towards self-explainable graph neural network. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, Queensland, Australia, pp. 302–311, 2021. DOI: 10.1145/3459637.3482306.

[30] T. Zhao, G. Liu, D. Wang, W. Yu, M. Jiang. Learning from counterfactual links for link prediction. In *Proceedings of the 39th International Conference on Machine Learning*, Baltimore, USA, pp. 26911–26926, 2022.

[31] Y. S. Dong, N. H. Liu, B. Jalaian, J. D. Li. EDITS: Modeling and mitigating data bias for graph neural networks. In *Proceedings of ACM Web Conference*, Lyon, France, pp. 1259–1269, 2022. DOI: 10.1145/3485447.3512173.

[32] Ö. D. Köse, Y. N. Shen. Fairness-aware node representation learning, [Online], Available: https://arxiv.org/abs/2106.05391, 2021.

[33] E. Y. Dai, T. X. Zhao, H. S. Zhu, J. J. Xu, Z. M. Guo, H. Liu, J. L. Tang, S. H. Wang. A comprehensive survey on trustworthy graph neural networks: Privacy, robustness, fairness, and explainability, [Online], Available: https://arxiv.org/abs/2204.08570, 2022.

[34] M. Kusner, J. R. Loftus, C. Russell, R. Silva. Counterfactual fairness. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, Long Beach, USA, pp. 4069–4079, 2017.

[35] S. Verma, V. Boonsanong, M. Hoang, K. E. Hines, J. P. Dickerson, C. Shah. Counterfactual explanations and algorithmic resourses for machine learning: A review, [Online], Available: https://arxiv.org/abs/2010.10596, 2020.

[36] S. Pitis, E. Creager, A. Garg. Counterfactual data augmentation using locally factored dynamics. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, Vancouver, Canada, pp. 3976–3990, 2020.

[37] L. Y. Yao, Z. X. Chu, S. Li, Y. L. Li, J. Gao, A. D. Zhang. A survey on causal inference. *ACM Transactions on Knowledge Discovery from Data*, vol. 15, no. 5, Article number 74, 2021. DOI: 10.1145/3444944.

[38] G. W. Imbens. Nonparametric estimation of average treatment effects under exogeneity: A review. *Review of Economics and Statistics*, vol. 86, no. 1, pp. 4–29, 2004. DOI: 10.1162/003465304323023651.

[39] R. H. Dehejia, S. Wahba. Causal effects in nonexperimental studies: Reevaluating the evaluation of training programs. *Journal of the American statistical Association*, vol. 94, no. 448, pp. 1053–1062, 1999. DOI: 10.1080/01621459.1999.10473858.

[40] Cross-Disorder Group of the Psychiatric Genomics Consortium. Identification of risk loci with shared effects on five major psychiatric disorders: A genome-wide analysis. *The Lancet*, vol. 381, no. 9875, pp. 1371–1379, 2013. DOI: 10.1016/S0140-6736(12)62129-1.

[41] R. C. Guo, L. Cheng, J. D. Li, P. R. Hahn, H. Liu. A survey of learning causality with data: Problems and methods. *ACM Computing Surveys*, vol. 53, no. 4, Article mumber 75, 2021. DOI: 10.1145/3397269.

[42] J. Kaddour, A. Lynch, Q. Liu, M. J. Kusner, R. Silva. Causal machine learning: A survey and open problems, [Online], Available: https://arxiv.org/abs/2206.15475, 2022.

[43] J. Pearl, M. Glymour, N. P. Jewell. *Causal Inference in Statistics: A Primer*, Chichester, USA: John Wiley & Sons, 2016.

[44] N. Mehrabi, F. Morstatter, N. Saxena, K. Lerman, A. Galstyan. A survey on bias and fairness in machine learning. *ACM Computing Surveys*, vol. 54, no. 6, Article number 115, 2022. DOI: 10.1145/3457607.

[45] D. V. Carvalho, E. M. Pereira, J. S. Cardoso. Machine learning interpretability: A survey on methods and metrics. *Electronics*, vol. 8, no. 8, Article number 832, 2019. DOI: 10.3390/electronics8080832.

[46] Z. Ying, D. Bourgeois, J. X. You, M. Zitnik, J. Leskovec. GNNExplainer: Generating explanations for graph neural networks. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, Vancouver, Canada, 2019.

[47] A. Lucic, M. A. Ter Hoeve, G. Tolomei, M. de Rijke, F. Silvestri. CF-GNNexplainer: Counterfactual explanations for graph neural networks. In *Proceedings of the 25th International Conference on Artificial Intelligence and Statistics*, pp. 4499–4511, 2022.

[48] K. Makhlouf, S. Zhioua, C. Palamidessi. Survey on causal-based machine learning fairness notions, [Online], Available: https://arxiv.org/abs/2010.09553, 2020.

[49] I. Stepin, J. M. Alonso, A. Catala, M. Pereira-Fariña. A survey of contrastive and counterfactual explanation generation methods for explainable artificial intelligence. *IEEE Access*, vol. 9, pp. 11974–12001, 2021. DOI: 10.1109/ACCESS.2021.3051315.

[50] A. Artelt, B. Hammer. On the computation of counterfactual explanations – A survey, [Online], Available: https://arxiv.org/abs/1911.07749, 2019.

[51] L. Oneto, N. Navarin, B. Biggio, F. Errica, A. Micheli, F. Scarselli, M. Bianchini, L. Demetrio, P. Bongini, A. Tacchella, A. Sperduti. Towards learning trustworthily, automatically, and with guarantees on graphs: An overview. *Neurocomputing*, vol. 493, pp. 217–243, 2022. DOI: 10.1016/J.NEUCOM.2022.04.072.

[52] M. A. Prado-Romero, B. Prenkaj, G. Stilo, F. Giannotti. A survey on graph counterfactual explanations: Definitions, methods, evaluation, [Online], Available: https://arxiv.org/abs/2210.12089, 2022.

[53] A. Feder, K. A. Keith, E. Manzoor, R. Pryzant, D. Sridhar, Z. Wood-Doughty, J. Eisenstein, J. Grimmer, R. Reichart, M. E. Roberts, B. M. Stewart, V. Veitch, D. Y. Yang. Causal inference in natural language processing: Estimation, prediction, interpretation and beyond. *Transactions of the Association for Computational Linguistics*, vol. 10, pp. 1138–1158, 2022. DOI: 10.1162/tacl_a_00511.

[54] L. Cheng, A. Mosallanezhad, P. Sheth, H. Liu. Causal learning for socially responsible AI. In *Proceedings of the 30th International Joint Conference on Artificial Intelligence*, Montreal, Canada, pp. 4374–4381, 2021. DOI: 10.24963/ijcai.2021/598.

[55] Y. Liu, Y. S. Wei, H. Yan, G. B. Li, L. Lin. Causal reasoning meets visual representation learning: A prospective study. *Machine Intelligence Research*, vol. 19, no. 6, pp. 485–511, 2022. DOI: 10.1007/S11633-022-1362-z.

[56] P. Sanchez, J. P. Voisey, T. Xia, H. I. Watson, A. Q. O'Neil, S. A. Tsaftaris. Causal machine learning for healthcare and precision medicine. *Royal Society Open Science*, vol. 9, no. 8, Article number 220638, 2022. DOI: 10.1098/rsos.220638.

[57] A. Vlontzos, D. Rueckert, B. Kainz. A review of causality for learning algorithms in medical image analysis, [Online], Available: https://arxiv.org/abs/2206.05498, 2022.

[58] F. L. Fan, J. J. Xiong, M. Z. Li, G. Wang. On interpretability of artificial neural networks: A survey. *IEEE Transactions on Radiation and Plasma Medical Sciences*, vol. 5, no. 6, pp. 741–760, 2021. DOI: 10.1109/trpms.2021.3066428.

[59] D. B. Rubin. Estimating causal effects of treatments in randomized and nonrandomized studies. *Journal of Educational Psychology*, vol. 66, no. 5, pp. 688–701, 1974. DOI: 10.1037/h0037350.

[60] D. B. Rubin. Causal inference using potential outcomes: Design, modeling, decisions. *Journal of the American Statistical Association*, vol. 100, no. 469, pp. 322–331, 2005. DOI: 10.1198/016214504000001880.

[61] E. T. Bullmore, D. S. Bassett. Brain graphs: Graphical models of the human brain connectome. *Annual Review of Clinical Psychology*, vol. 7, pp. 113–140, 2011. DOI: 10.1146/annurev-clinpsy-040510-143934.

[62] R. Ying, R. N. He, K. F. Chen, P. Eksombatchai, W. L. Hamilton, J. Leskovec. Graph convolutional neural networks for web-scale recommender systems. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, London, UK, pp. 974–983, 2018. DOI: 10.1145/3219819.3219890.

[63] M. H. Zhang, Z. C. Cui, M. Neumann, Y. X. Chen. An end-to-end deep learning architecture for graph classification. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence*, New Orleans, USA, pp. 4438–4445, 2018. DOI: 10.1609/aaai.v32i1.11782.

[64] S. Bhagat, G. Cormode, S. Muthukrishnan. Node classification in social networks. *Social Network Data Analytics*, C. C. Aggarwal, Ed., New York, USA: Springer, pp. 115–148, 2011. DOI: 10.1007/978-1-4419-8462-3_5.

[65] S. V. N. Vishwanathan, N. N. Schraudolph, R. Kondor, K. M. Borgwardt. Graph kernels. *The Journal of Machine Learning Research*, vol. 11, pp. 1201–1242, 2010.

[66] D. Liben-Nowell, J. Kleinberg. The link-prediction problem for social networks. *Journal of the American Society for Information Science and Technology*, vol. 58, no. 7, pp. 1019–1031, 2007. DOI: 10.1002/ASI.20591.

[67] N. Pržulj. Biological network comparison using graphlet degree distribution. *Bioinformatics*, vol. 23, no. 2, pp. e177–e183, 2007. DOI: 10.1093/BIOINFORMATICS/BTL301.

[68] J. Z. Qiu, Y. X. Dong, H. Ma, J. Li, K. S. Wang, J. Tang. Network embedding as matrix factorization: Unifying DeepWalk, LINE, PTE, and node2vec. In *Proceedings of the 11th ACM International Conference on Web Search and Data Mining*, Marina Del Rey, USA, pp. 459–467, 2018. DOI: 10.1145/3159652.3159706.

[69] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P.

Liò, Y. Bengio. Graph attention networks. In *Proceedings of the 6th International Conference on Learning Representations*, Vancouver, Canada, 2018.

[70] T. Xiao, Z. Y. Chen, D. L. Wang, S. H. Wang. Learning how to propagate messages in graph neural networks. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pp. 1894–1903, 2021. DOI: 10.1145/3447548.3467451.

[71] J. J. Xu, E. Y. Dai, X. Zhang, S. H. Wang. HP-GMN: Graph memory networks for heterophilous graphs. In *Proceedings of IEEE International Conference on Data Mining*, Orlando, USA, pp. 1263–1268, 2022. doi: 10.1109/ICDM54844.2022.00165.

[72] T. X. Zhao, D. S. Luo, X. Zhang, S. H. Wang. Topoimb: Toward topology-level imbalance in learning from graphs. In *Proceedings of the 1st Learning on Graphs Conference*, Article number 37, 2022.

[73] Z. W. Zhang, M. H. Lin, E. Y. Dai, S. H. Wang. Rethinking graph backdoor attacks: A distribution-preserving perspective, [Online], Available: https://arxiv.org/abs/2405.10757, 2024.

[74] M. H. Lin, T. Xiao, E. Y. Dai, X. Zhang, S. H. Wang. Certifiably robust graph contrastive learning. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, New Orleans, USA, 2023.

[75] E. Y. Dai, C. Aggarwal, S. H. Wang. NRGNN: Learning a label noise resistant graph neural network on sparsely and noisily labeled graphs. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, Singapore, pp. 227–236, 2021. DOI: 10.1145/3447548.3467364.

[76] T. X. Zhao, X. Zhang, S. H. Wang. GraphSMOTE: Imbalanced node classification on graphs with graph neural networks. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*, pp. 833–841, 2021. DOI: 10.1145/3437963.3441720.

[77] E. Y. Dai, W. Jin, H. Liu, S. H. Wang. Towards robust graph neural networks for noisy graphs with sparse labels. In *Proceedings of the 15th ACM International Conference on Web Search and Data Mining*, Arizona, USA, pp. 181–191, 2022. DOI: 10.1145/3488560.3498408.

[78] F. L. Wang, T. X. Zhao, S. H. Wang. Distribution consistency based self-training for graph neural networks with sparse labels. In *Proceedings of the 17th ACM International Conference on Web Search and Data Mining*, Merida, Mexico, pp. 712–720, 2024. DOI: 10.1145/3616855.3635793.

[79] N. M. Nasrabadi. Pattern recognition and machine learning. *Journal of Electronic Imaging*, vol. 16, no. 4, Article number 049901, 2007. DOI: 10.1117/1.2819119.

[80] E. Z. Liu, B. Haghgoo, A. S. Chen, A. Raghunathan, P. W. Koh, S. Sagawa, P. Liang, C. Finn. Just train twice: Improving group robustness without training group information. In *Proceedings of the 38th International Conference on Machine Learning*, pp. 6781–6792, 2021.

[81] S. Sagawa, P. W. Koh, T. B. Hashimoto, P. Liang. Distributionally robust neural networks. In *Proceedings of the 8th International Conference on Learning Representations*, Addis Ababa, Ethiopia, 2020.

[82] M. Arjovsky, L. Bottou, I. Gulrajani, D. Lopez-Paz. Invariant risk minimization, [Online], Available: https://arxiv.org/abs/1907.02893, 2019.

[83] J. Mitrovic, B. McWilliams, J. C. Walker, L. H. Buesing, C. Blundell. Representation learning via invariant causal mechanisms. In *Proceedings of the 9th International Conference on Learning Representations*, 2021.

[84] J. Pearl. Causal inference in statistics: An overview. *Statistics Surveys*, vol. 3, pp. 96–146, 2009. DOI: 10.1214/09-SS057.

[85] B. Schölkopf, F. Locatello, S. Bauer, N. R. Ke, N. Kalchbrenner, A. Goyal, Y. Bengio. Towards causal representation learning, [Online], Available: https://arxiv.org/abs/2102.11107, 2021.

[86] J. D. Wang, C. L. Lan, C. Liu, Y. D. Ouyang, T. Qin. Generalizing to unseen domains: A survey on domain generalization. In *Proceedings of the 30th International Joint Conference on Artificial Intelligence*, Montreal, Canada, pp. 4627–4635, 2021. DOI: 10.24963/ijcai.2021/628.

[87] B. Scholkopf, J. von Kügelgen. From statistical to causal learning, [Online], Available: https://arxiv.org/abs/2204.00607, 2022.

[88] R. Shanmugam. Elements of causal inference: Foundations and learning algorithms. *Journal of Statistical Computation and Simulation*, vol. 88, no. 16, pp. 3248–3248, 2018. DOI: 10.1080/00949655.2018.1505197.

[89] J. Ma, R. C. Guo, M. T. Wan, L. Q. Yang, A. D. Zhang, J. D. Li. Learning fair node representations with graph counterfactual fairness. In *Proceedings of the 15th ACM International Conference on Web Search and Data Mining*, Arizona, USA, pp. 695–703, 2022. DOI: 10.1145/3488560.3498391.

[90] J. H. Chen. Fair lending needs explainable models for responsible recommendation, [Online], Available: https://arxiv.org/abs/1809.04684, 2018.

[91] Y. S. Dong, J. Ma, S. Wang, C. Chen, J. D. Li. Fairness in graph mining: A survey. *IEEE Transactions on Knowledge and Data Engineering*, vol. 35, no. 10, pp. 10583–10602, 2023. DOI: 10.1109/TKDE.2023.3265598.

[92] C. Agarwal, H. Lakkaraju, M. Zitnik. Towards a unified framework for fair and stable graph representation learning. In *Proceedings of the 37th Conference on Uncertainty in Artificial Intelligence*, pp. 2114–2124, 2021.

[93] E. Y. Dai, S. H. Wang. Learning fair graph neural networks with limited and private sensitive attribute information. *IEEE Transactions on Knowledge and Data Engineering*, vol. 35, no. 7, pp. 7103–7117, 2023. DOI: 10.1109/TKDE.2022.3197554.

[94] X. Zhang, L. Zhang, B. Jin, X. J. Lu. A multi-view confidence-calibrated framework for fair and stable graph representation learning. In *Proceedings of IEEE International Conference on Data Mining*, Auckland, New Zealand, pp. 1493–1498, 2021. doi: 10.1109/ICDM51629.2021.00194.

[95] H. S. Zhu, E. Y. Dai, H. Liu, S. H. Wang. Learning fair models without sensitive attributes: A generative approach. *Neurocomputing*, vol. 561, Article number 126841, 2023. DOI: 10.1016/j.neucom.2023.126841.

[96] A. Jalal, S. Karmalkar, J. Hoffmann, A. Dimakis, E. Price. Fairness for image generation with uncertain sensitive attributes. In *Proceedings of the 38th International*

*Conference on Machine Learning*, pp. 4721–4732, 2021.

[97] S. Garg, V. Perot, N. Limtiaco, A. Taly, E. H. Chi, A. Beutel. Counterfactual fairness in text classification through robustness. In *Proceedings of AAAI/ACM Conference on AI, Ethics, and Society*, Honolulu, USA, pp. 219–226, 2019. DOI: 10.1145/3306618.3317950.

[98] V. Nathan, V. Sivaraman, R. Addanki, M. Khani, P. Goyal, M. Alizadeh. End-to-end transport for video QoE fairness. In *Proceedings of ACM Special Interest Group on Data Communication*, Beijing, China, pp. 408–423, 2019. DOI: 10.1145/3341302.3342077.

[99] P. J. Bickel, E. A. Hammel, J. W. O'Connell. Sex bias in graduate admissions: Data from Berkeley: Measuring bias is harder than is usually assumed, and the evidence is sometimes contrary to expectation. *Science*, vol. 187, no. 4175, pp. 398–404, 1975. DOI: 10.1126/science.187.4175.398.

[100] F. Masrour, T. Wilson, H. Yan, P. N. Tan, A. Esfahanian. Bursting the filter bubble: Fairness-aware network link prediction. In *Proceedings of the 34th AAAI Conference on Artificial Intelligence*, New York, USA, pp. 841–848, 2020. DOI: 10.1609/AAAI.V34I01.5429.

[101] A. Bose, W. Hamilton. Compositional fairness constraints for graph embeddings. In *Proceedings of the 36th International Conference on Machine Learning*, Long Beach, USA, pp. 715–724, 2019.

[102] P. Z. Li, Y. F. Wang, H. Zhao, P. Y. Hong, H. F. Liu. On dyadic fairness: Exploring and mitigating bias in graph connections. In *Proceedings of the 9th International Conference on Learning Representations*, 2021.

[103] N. Wang, L. Lin, J. D. Li, H. N. Wang. Unbiased graph embedding with biased graph observations. In *Proceedings of ACM Web Conference*, Lyon, France, pp. 1423–1433, 2022. DOI: 10.1145/3485447.3512189.

[104] A. Khajehnejad, M. Khajehnejad, M. Babaei, K. P. Gummadi, A. Weller, B. Mirzasoleiman. CrossWalk: Fairness-enhanced node representation learning. In *Proceedings of the 36th AAAI Conference on Artificial Intelligence*, pp. 11963–11970, 2022. DOI: 10.1609/AAAI.V36I11.21454.

[105] T. Rahman, B. Surma, M. Backes, Y. Zhang. Fairwalk: Towards fair graph embedding. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, Macao, China, pp. 3289–3295, 2019. DOI: 10.24963/IJCAI.2019/456.

[106] Z. M. Guo, J. L. Li, T. Xiao, Y. Ma, S. H. Wang. Towards fair graph neural networks via graph counterfactual. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, Birmingham, UK, pp. 669–678, 2023. DOI: 10.1145/3583780.3615092.

[107] H. Y. Ling, Z. M. Jiang, Y. Z. Luo, S. W. Ji, N. Zou. Learning fair graph representations via automated data augmentations. In *Proceedings of the 11th International Conference on Learning Representations*, Kigali, Rwanda, 2023.

[108] K. Z. Ding, Z. Xu, H. H. Tong, H. Liu. Data augmentation for deep graph learning: A survey. *ACM SIGKDD Explorations Newsletter*, vol. 24, no. 2, pp. 61–77, 2022. DOI: 10.1145/3575637.3575646.

[109] Z. M. Jiang, X. T. Han, C. Fan, Z. R. Liu, N. Zou, A. Mostafavi, X. Hu. FMP: Toward fair graph message passing against topology bias, [Online], Available: https://arxiv.org/abs/2202.04187, 2022.

[110] C. Agarwal, M. Zitnik, H. Lakkaraju. Probing GNN explainers: A rigorous theoretical and empirical analysis of GNN explanation methods. In *Proceedings of the 25th International Conference on Artificial Intelligence and Statistics*, pp. 8969–8996, 2022.

[111] Z. C. Wang, G. Narasimhan, X. Yao, W. B. Zhang. Mitigating multisource biases in graph neural networks via real counterfactual samples. In *Proceedings of IEEE International Conference on Data Mining*, Shanghai, China, pp. 638–647, 2023. DOI: 10.1109/ICDM58522.2023.00073.

[112] Z. C. Wang, M. K. Qiu, M. Chen, M. B. Salem, X. Yao, W. B. Zhang. Toward fair graph neural networks via real counterfactual samples. *Knowledge and Information Systems*, vol. 66, pp. 6617–6641, 2024. DOI: 10.1007/s10115-024-02161-z.

[113] Y. Zhang, D. Z. Cheng, T. S. Yao, X. Y. Yi, L. C. Hong, E. H. Chi. A model of two tales: Dual transfer learning framework for improved long-tail item recommendation. In *Proceedings of Web Conference*, Ljubljana, Slovenia, pp. 2220–2231, 2021. DOI: 10.1145/3442381.3450086.

[114] C. Dwork, M. Hardt, T. Pitassi, O. Reingold, R. Zemel. Fairness through awareness. In *Proceedings of the 3rd Innovations in Theoretical Computer Science*, Cambridge, USA, pp. 214–226, 2012. DOI: 10.1145/2090236.2090255.

[115] M. Hardt, E. Price, N. Srebro. Equality of opportunity in supervised learning. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, Barcelona, Spain, pp. 3323–3331, 2016.

[116] L. Takac, M. Zabovsky. Data analysis in public social networks. In *Proceedings of International Scientific Conference & International Workshop Present Day Trends of Innovations*, Lomza, Poland, 2012.

[117] I. C. Yeh, C. H. Lien. The comparisons of data mining techniques for the predictive accuracy of probability of default of credit card clients. *Expert Systems with Applications*, vol. 36, no. 2, pp. 2473–2480, 2009. DOI: 10.1016/j.eswa.2007.12.020.

[118] A. Asuncion, D. J. Newman. UCI machine learning repository, Irvine University of California, Irvine, USA, 2007.

[119] K. L. Jordan, T. L. Freiburger. The effect of race/ethnicity on sentencing: Examining sentence type, jail length, and prison length. *Journal of Ethnicity in Criminal Justice*, vol. 13, no. 3, pp. 179–196, 2015. DOI: 10.1080/15377938.2014.984045.

[120] F. Noé, A. Tkatchenko, K. R. Müller, C. Clementi. Machine learning for molecular simulation. *Annual Review of Physical Chemistry*, vol. 71, pp. 361–390, 2020. DOI: 10.1146/annurev-physchem-042018-052331.

[121] A. Barredo Arrieta, N. Díaz-Rodríguez, J. Del Ser, A. Bennetot, S. Tabik, A. Barbado, S. Garcia, S. Gil-Lopez, D. Molina, R. Benjamins, R. Chatila, F. Herrera. Explainable artificial intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. *Information Fusion*, vol. 58, pp. 82–115, 2020. DOI: 10.1016/j.inffus.2019.12.012.

[122] T. Martin. Interpretable Machine Learning, Master dissertation, University of Cambridge, UK, 2019.

[123] H. Zhang, B. Wu, X. L. Yuan, S. R. Pan, H. H. Tong, J. Pei. Trustworthy graph neural networks: Aspects, methods, and trends. *Proceedings of the IEEE*, vol. 112, no. 2, pp. 97–139, 2024. DOI: 10.1109/JPROC.2024.3369017.

[124] T. X. Zhao, D. S. Luo, X. Zhang, S. H. Wang. Faithful and consistent graph neural network explanations with rationale alignment. *ACM Transactions on Intelligent Systems and Technology*, vol. 14, no. 5, Article number 92, 2023. DOI: 10.1145/3616542.

[125] T. X. Zhao, D. S. Luo, X. Zhang, S. H. Wang. Towards faithful and consistent explanations for graph neural networks. In *Proceedings of the 16th ACM International Conference on Web Search and Data Mining*, Singapore, pp. 634–642, 2023. DOI: 10.1145/3539597.3570421.

[126] H. X. Cai, H. M. Zhang, D. C. Zhao, J. X. Wu, L. Wang. FP-GNN: A versatile deep learning architecture for enhanced molecular property prediction. *Briefings in Bioinformatics*, vol. 23, no. 6, Artile number bbac408, 2022. DOI: 10.1093/BIB/BBAC408.

[127] H. Yuan, H. Y. Yu, S. R. Gui, S. W. Ji. Explainability in graph neural networks: A taxonomic survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 45, no. 5, pp. 5782–5799, 2023. DOI: 10.1109/TPAMI.2022.3204236.

[128] D. S. Luo, W. Cheng, D. K. Xu, W. C. Yu, B. Zong, H. F. Chen, X. Zhang. Parameterized explainer for graph neural network. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, Vancouver, Canada, 2020.

[129] M. Bajaj, L. Y. Chu, Z. Y. Xue, J. Pei, L. J. Wang, P. C. H. Lam, Y. Zhang. Robust counterfactual explanations on graph neural networks. In *Proceedings of 35th International Conference on Neural Information Processing Systems*, 2021.

[130] J. T. Tan, S. J. Geng, Z. H. Fu, Y. Q. Ge, S. Y. Xu, Y. Q. Li, Y. F. Zhang. Learning and evaluating graph neural network explanations based on counterfactual and factual reasoning. In *Proceedings of ACM Web Conference*, Lyon, France, pp. 1018–1027, 2022. DOI: 10.1145/3485447.3511948.

[131] Y. Sun, A. Valente, S. J. Liu, D. K. Wang. Preserve, promote, or attack? GNN explanation via topology perturbation, [Online], Available: https://arxiv.org/abs/2103.13944, 2021.

[132] C. Abrate, F. Bonchi. Counterfactual graphs for explainable classification of brain networks. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, Singapore, pp. 2495–2504, 2021. DOI: 10.1145/3447548.3467154.

[133] J. Ma, R. C. Guo, S. Mishra, A. D. Zhang, J. Li. CLEAR: Generative counterfactual explanations on graphs. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, New Orleans, USA, 2022.

[134] Z. X. Huang, M. Kosan, S. Medya, S. Ranu, A. Singh. Global counterfactual explainer for graph neural networks. In *Proceedings of the 16th ACM International Conference on Web Search and Data Mining*, Singapore, pp. 141–149, 2023. DOI: 10.1145/3539597.3570376.

[135] Y. F. Liu, C. Chen, Y. Z. Liu, X. Zhang, S. H. Xie. Multi-objective explanations of GNN predictions. In *Proceedings of IEEE International Conference on Data Mining*, Auckland, New Zealand, pp. 409–418, 2021. DOI: 10.1109/ICDM51629.2021.00052.

[136] R. C. Cai, Y. X. Zhu, X. X. Chen, Y. Fang, M. Wu, J. Qiao, Z. F. Hao. On the probability of necessity and sufficiency of explaining graph neural networks: A lower bound optimization approach, [Online], Available: https://arxiv.org/abs/2212.07056, 2022.

[137] L. Ohly. Flow-based Counterfactuals for Interpretable Graph Node Classification, Bachelor's dissertation, Freien University, Berlin, Germany, 2022.

[138] M. A. Prado-Romero, B. Prenkaj, G. Stilo, A. Celi, E. Estevanell-Valladares, D. A. Valdés-Pérez. Ensemble approaches for graph counterfactual explanations. In *Proceedings of the 3rd Italian Workshop on Explainable Artificial Intelligence Co-Located with 21th International Conference of the Italian Association for Artificial Intelligence*, Udine, Italy, pp. 88–97, 2022.

[139] H. Yuan, J. L. Tang, X. Hu, S. W. Ji. XGNN: Towards model-level explanations of graph neural networks. In *Proceedings of the 26th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pp. 430–438, 2020. DOI: 10.1145/3394486.3403085.

[140] F. Baldassarre, H. Azizpour. Explainability techniques for graph convolutional networks, [Online], Available: https://arxiv.org/abs/1905.13686, 2019.

[141] Q. Huang, M. Yamada, Y. Tian, D. Singh, Y. Chang. GraphLIME: Local interpretable model explanations for graph neural networks. *IEEE Transactions on Knowledge and Data Engineering*, vol. 35, no. 7, pp. 6968–6972, 2023. DOI: 10.1109/TKDE.2022.3187455.

[142] S. Q. Miao, M. Liu, P. Li. Interpretable and generalizable graph learning via stochastic attention mechanism. In *Proceedings of the 39th International Conference on Machine Learning*, Baltimore, USA, pp. 15524–15543, 2022.

[143] E. Y. Dai, S. H. Wang. Towards prototype-based self-explainable graph neural network, [Online], Available: https://arxiv.org/abs/2210.01974, 2022.

[144] W. Y. Lin, H. Lan, B. C. Li. Generative causal explanations for graph neural networks. In *Proceedings of the 38th International Conference on Machine Learning*, pp. 6666–6679, 2021.

[145] S. Joshi, O. Koyejo, W. Vijitbenjaronk, B. Kim, J. Ghosh. Towards realistic individual recourse and actionable explanations in black-box decision making systems, [Online], Available: https://arxiv.org/abs/1907.09615, 2019.

[146] M. A. Prado-Romero, G. Stilo. GRETEL: A unified framework for graph counterfactual explanation evaluation, [Online], Available: https://arxiv.org/abs/2206.02957, 2022.

[147] C. Chhablani, S. Jain, A. Channesh, I. A. Kash, S. Medya. Game-theoretic counterfactual explanation for graph neural networks. In *Proceedings of ACM on Web Conference*, Singapore, pp. 503–514, 2024. DOI: 10.1145/3589334.3645419.

[148] F. Hvilshøj, A. Iosifidis, I. Assent. ECINN: Efficient counterfactuals from invertible neural networks. In *Proceedings of the 32nd British Machine Vision Conference*, Article number 43, 2021.

[149] P. Dubey, L. S. Shapley. Mathematical properties of the

Banzhaf power index. *Mathematics of Operations Research*, vol. 4, no. 2, pp. 99–131, 1979. DOI: 10.1287/moor.4.2.99.

[150] J. F. Banzhaf III. Weighted voting Doesn't work: A mathematical analysis. *Rutgers Law Review*, vol. 19, no. 2, pp. 317–343, 1965.

[151] D. P. Kingma, M. Welling. Auto-encoding variational Bayes. In *Proceedings of the 2nd International Conference on Learning Representations*, Banff, Canada, 2014. DOI: 10.48550/arXiv.1312.6114.

[152] A. K. Debnath, R. L. Lopez De Compadre, G. Debnath, A. J. Shusterman, C. Hansch. Structure-activity relationship of mutagenic aromatic and heteroaromatic nitro compounds. Correlation with molecular orbital energies and hydrophobicity. *Journal of Medicinal Chemistry*, vol. 34, no. 2, pp. 786–797, 1991. DOI: 10.1021/jm00106a046.

[153] N. Wale, I. A. Watson, G. Karypis. Comparison of descriptor spaces for chemical compound retrieval and classification. *Knowledge and Information Systems*, vol. 14, no. 3, pp. 347–375, 2008. DOI: 10.1007/s10115-007-0103-5.

[154] K. Kersting, N. M. Kriege, C. Morris, P. Mutzel, M. Neumann. Benchmark data sets for graph kernels, [Online], Available: http://graphkernels.cs.tu-dortmund.de, 2016.

[155] C. Craddock, Y. Benhajali, C. Chu, F. Chouinard, A. Evans, A. Jakab, B. S. Khundrakpam, J. D. Lewis, Q. Y. Li, M. Milham, C. G. Yan, P. Bellec. The neuro bureau preprocessing initiative: Open sharing of preprocessed neuroimaging data and derivatives. *Frontiers in Neuroinformatics*, vol. 7, Article number 27, 2013. DOI: 10.3389/conf.fninf.2013.09.00041.

[156] J. A. Brown, J. D. Rudie, A. Bandrowski, J. D. Van Horn, S. Y. Bookheimer. The UCLA multimodal connectivity database: A web-based platform for brain connectivity matrix sharing and analysis. *Frontiers in Neuroinformatics*, vol. 6, Article number 28, 2012. DOI: 10.3389/fninf.2012.00028.

[157] A. Kumar, S. S. Singh, K. Singh, B. Biswas. Link prediction techniques, applications, and performance: A survey. *Physica A: Statistical Mechanics and its Applications*, vol. 553, Article number 124289, 2020. DOI: 10.1016/j.physa.2020.124289.

[158] W. J. Wang, F. L. Feng, X. N. He, H. W. Zhang, T. S. Chua. Clicks can be cheating: Counterfactual recommendation for mitigating clickbait issue. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 1288–1297, 2021. DOI: 10.1145/3404835.3462962.

[159] Z. H. Chen, F. Silvestri, J. Wang, Y. F. Zhang, Z. H. Huang, H. Ahn, G. Tolomei. GREASE: Generate factual and counterfactual explanations for GNN-based recommendations, [Online], Available: https://arxiv.org/abs/2208.04222, 2022.

[160] S. L. Mu, Y. L. Li, W. X. Zhao, J. Y. Wang, B. L. Ding, J. R. Wen. Alleviating spurious correlations in knowledge-aware recommendations through counterfactual generator. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, Madrid, Spain, pp. 1401–1411, 2022. DOI: 10.1145/3477495.3531934.

[161] C. Liu, C. Gao, D. P. Jin, Y. Li. Improving location recommendation with urban knowledge graph, [Online], Available: https://arxiv.org/abs/2111.01013, 2021.

[162] W. Z. Song, S. J. Wang, Y. Wang, K. P. Liu, X. Y. Liu, M. H. Yin. A counterfactual collaborative session-based recommender system. In *Proceedings of ACM Web Conference*, Austin, USA, pp. 971–982, 2023. DOI: 10.1145/3543507.3583321.

[163] H. Chang, J. Cai, J. Li. Knowledge graph completion with counterfactual augmentation. In *Proceedings of ACM Web Conference*, Austin, USA, pp. 2611–2620, 2023. DOI: 10.1145/3543507.3583401.

[164] W. Chen, Y. Q. Wu, Z. Zhang, F. Z. Zhuang, Z. S. He, R. B. Xie, F. Xia. FairGap: Fairness-aware recommendation via generating counterfactual graph. *ACM Transactions on Information Systems*, vol. 42, no. 4, Article number 94, 2024. DOI: 10.1145/3638352.

[165] L. Boratto, F. Fabbri, G. Fenu, M. Marras, G. Medda. Counterfactual graph augmentation for consumer unfairness mitigation in recommender systems. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, Birmingham, UK, pp. 3753–3757, 2023. DOI: 10.1145/3583780.3615165.

[166] N. Barbieri, F. Bonchi, G. Manco. Who to follow and why: Link prediction with explanations. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, New York, USA, pp. 1266–1275, 2014. DOI: 10.1145/2623330.2623733.

[167] Z. Q. Sun, Z. H. Deng, J. Y. Nie, J. Tang. RotatE: Knowledge graph embedding by relational rotation in complex space. In *Proceedings of the 7th International Conference on Learning Representations*, New Orleans, USA, 2019.

[168] K. Abbas, A. Abbasi, S. Dong, L. Niu, L. H. Yu, B. L. Chen, S. M. Cai, Q. Hasan. Application of network link prediction in drug discovery. *BMC Bioinformatics*, vol. 22, no. 1, Article number 187, 2021. DOI: 10.1186/s12859-021-04082-y.

[169] L. Katz. A new status index derived from sociometric analysis. *Psychometrika*, vol. 18, no. 1, pp. 39–43, 1953. DOI: 10.1007/BF02289026.

[170] H. Nassar, A. R. Benson, D. F. Gleich. Pairwise link prediction. In *Proceedings of IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, Vancouver, Canada, pp. 386–393, 2019. DOI: 10.1145/3341161.3342897.

[171] T. N. Kipf, M. Welling. Variational graph auto-encoders, [Online], Available: https://arxiv.org/abs/1611.07308, 2016.

[172] L. Y. Lü, T. Zhou. Link prediction in complex networks: A survey. *Physica A: Statistical Mechanics and its Applications*, vol. 390, no. 6, pp. 1150–1170, 2011. DOI: 10.1016/j.physa.2010.11.027.

[173] V. D. Blondel, J. L. Guillaume, R. Lambiotte, E. Lefebvre. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, vol. 2008, no. 10, Article number P10008, 2008. DOI: 10.1088/1742-5468/2008/10/P10008.

[174] K. Hassani, A. H. Khasahmadi. Contrastive multi-view representation learning on graphs. In *Proceedings of the 37th International Conference on Machine Learning*, pp. 4116–4126, 2020.

[175] S. X. Ji, S. R. Pan, E. Cambria, P. Marttinen, P. S. Yu. A

survey on knowledge graphs: Representation, acquisition, and applications. *IEEE Transactions on Neural Networks and Learning Systems*, vol. 33, no. 2, pp. 494–514, 2022. DOI: 10.1109/TNNLS.2021.3070843.

[176] Z. Chen, Y. H. Wang, B. Zhao, J. Cheng, X. Zhao, Z. T. Duan. Knowledge graph completion: A review. *IEEE Access*, vol. 8, pp. 192435–192456, 2020. DOI: 10.1109/ACCESS.2020.3030076.

[177] Y. Wang, Z. W. Liu, Z. W. Fan, L. C. Sun, P. S. Yu. DSKReG: Differentiable sampling on knowledge graph for recommendation with relational GNN. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, pp. 3513–3517, 2021. DOI: 10.1145/3459637.3482092.

[178] M. Schlichtkrull, T. N. Kipf, P. Bloem, R. van den Berg, I. Titov, M. Welling. Modeling relational data with graph convolutional networks. In *Proceedings of the 15th International Conference on the Semantic Web*, Heraklion, Greece, pp. 593–607, 2018. DOI: 10.1007/978-3-319-93417-4_38.

[179] Z. C. Zhu, Z. B. Zhang, L. P. Xhonneux, J. Tang. Neural bellman-ford networks: A general graph neural network framework for link prediction. In *Proceedings of the 35th International Conference on Neural Information Processing Systems*, 2021.

[180] N. Lim, B. Hooi, S. K. Ng, X. O. Wang, Y. L. Goh, R. R. Weng, J. Varadarajan. STP-UDGAT: Spatial-temporal-preference user dimensional graph attention network for next POI recommendation. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, pp. 845–854, 2020. DOI: 10.1145/3340531.3411876.

[181] X. C. Li, X. Wang, X. N. He, L. Chen, J. Xiao, T. S. Chua. Hierarchical fashion graph network for personalized outfit recommendation. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 159–168, 2020. DOI: 10.1145/3397271.3401080.

[182] Z. W. Liu, L. Meng, F. Jiang, J. W. Zhang, P. S. Yu. Deoscillated graph collaborative filtering, [Online], Available: https://arxiv.org/abs/2011.02100, 2020.

[183] G. Linden, B. Smith, J. York. Amazon.com recommendations: Item-to-item collaborative filtering. *IEEE Internet Computing*, vol. 7, no. 1, pp. 76–80, 2003. DOI: 10.1109/MIC.2003.1167344.

[184] B. M. Sarwar, G. Karypis, J. Konstan, J. Riedl. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th International Conference on World Wide Web*, Hong Kong, China, pp. 285–295, 2001. DOI: 10.1145/371920.372071.

[185] H. Wang, N. Y. Wang, D. Y. Yeung. Collaborative deep learning for recommender systems. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Sydney, Australia, pp. 1235–1244, 2015. DOI: 10.1145/2783258.2783273.

[186] S. Sedhain, A. K. Menon, S. Sanner, L. X. Xie. AutoRec: Autoencoders meet collaborative filtering. In *Proceedings of the 24th International Conference on World Wide Web*, Florence, Italy, pp. 111–112, 2015. DOI: 10.1145/2740908.2742726.

[187] Y. Koren, S. Rendle, R. Bell. Advances in collaborative filtering. *Recommender Systems Handbook*, F. Ricci, L. Rokach, B. Shapira, Eds., New York, USA: Springer, pp. 91–142, 2022. DOI: 10.1007/978-1-0716-2197-4_3.

[188] Y. Koren, R. Bell, C. Volinsky. Matrix factorization techniques for recommender systems. *Computer*, vol. 42, no. 8, pp. 30–37, 2009. DOI: 10.1109/MC.2009.263.

[189] T. Xiao, S. S. Liang, W. Z. Shen, Z. Q. Meng. Bayesian deep collaborative matrix factorization. In *Proceedings of the 33rd AAAI Conference on Artificial Intelligence*, Honolulu, USA, pp. 5474–5481, 2019. DOI: 10.1609/aaai.v33i01.33015474.

[190] S. Zhang, L. Yao, A. X. Sun, Y. Tay. Deep learning based recommender system: A survey and new perspectives. *ACM Computing Surveys*, vol. 52, no. 1, Article number 5, 2020. DOI: 10.1145/3285029.

[191] Q. T. Wu, H. R. Zhang, X. F. Gao, P. He, P. Weng, H. Gao, G. H. Chen. Dual graph attention networks for deep latent representation of multifaceted social effects in recommender systems. In *Proceedings of World Wide Web Conference*, San Francisco, USA, pp. 2091–2102, 2019. DOI: 10.1145/3308558.3313442.

[192] H. W. Wang, F. Z. Zhang, M. D. Zhang, J. Leskovec, M. Zhao, W. J. Li, Z. Y. Wang. Knowledge-aware graph neural networks with label smoothness regularization for recommender systems. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, Anchorage, USA, pp. 968–977, 2019. DOI: 10.1145/3292500.3330836.

[193] W. Q. Fan, Y. Ma, Q. Li, Y. He, E. Zhao, J. L. Tang, D. W. Yin. Graph neural networks for social recommendation. In *Proceedings of World Wide Web Conference*, San Francisco, USA, pp. 417–426, 2019. DOI: 10.1145/3308558.3313488.

[194] C. Gao, Y. Zheng, W. J. Wang, F. L. Feng, X. N. He, Y. Li. Causal inference in recommender systems: A survey and future directions. *ACM Transactions on Information Systems*, vol. 42, no. 4, Article number 88, 2024. DOI: 10.1145/3639048.

[195] T. Xiao, Z. Y. Chen, S. H. Wang. Representation matters when learning from biased feedback in recommendation. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, Atlanta, USA, pp. 2220–2229, 2022. DOI: 10.1145/3511808.3557431.

[196] T. Xiao, S. H. Wang. Towards unbiased and robust causal ranking for recommender systems. In *Proceedings of the 15th ACM International Conference on Web Search and Data Mining*, pp. 1158–1167, 2022. DOI: 10.1145/3488560.3498521.

[197] G. R. Zhou, X. Q. Zhu, C. R. Song, Y. Fan, H. Zhu, X. Ma, Y. H. Yan, J. Q. Jin, H. Li, K. Gai. Deep interest network for click-through rate prediction. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, London, UK, pp. 1059–1068, 2018. DOI: 10.1145/3219819.3219823.

[198] W. N. Zhang, J. R. Qin, W. Guo, R. M. Tang, X. Q. He. Deep learning for click-through rate estimation. In *Proceedings of the 30th International Joint Conference on Artificial Intelligence*, Montreal, Canada, pp. 4695–4703, 2021. DOI: 10.24963/ijcai.2021/636.

[199] Y. W. Wei, X. Wang, L. Q. Nie, X. N. He, R. C. Hong, T. S. Chua. MMGCN: Multi-modal graph convolution network for personalized recommendation of micro-video. In

*Proceedings of the 27th ACM International Conference on Multimedia*, Nice, France, pp. 1437–1445, 2019. DOI: 10.1145/3343031.3351034.

[200] S. S. Feng, X. T. Li, Y. F. Zeng, G. Cong, Y. M. Chee, Q. Yuan. Personalized ranking metric embedding for next new POI recommendation. In *Proceedings of the 24th International Joint Conference on Artificial Intelligence*, Buenos Aires, Argentina, pp. 2069–2075, 2015.

[201] H. Z. Yin, W. Q. Wang, H. Wang, L. Chen, X. F. Zhou. Spatial-aware hierarchical collaborative deep learning for POI recommendation. *IEEE Transactions on Knowledge and Data Engineering*, vol. 29, no. 11, pp. 2537–2551, 2017. DOI: 10.1109/TKDE.2017.2741484.

[202] S. L. Zhao, I. King, M. R. Lyu. A survey of point-of-interest recommendation in location-based social networks, [Online], Available: https://arxiv.org/abs/1607.00647, 2016.

[203] R. S. Sutton, A. G. Barto. *Reinforcement Learning: An Introduction*, Cambridge, USA: MIT Press, Article number 1054, 1998.

[204] F. Baradel, N. Neverova, J. Mille, G. Mori, C. Wolf. CoPhy: Counterfactual learning of physical dynamics. In *Proceedings of the 8th International Conference on Learning Representations*, Addis Ababa, Ethiopia, 2020.

[205] Y. Z. Li, A. Torralba, A. Anandkumar, D. Fox, A. Garg. Causal discovery in physical systems from videos. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, Vancouver, Canada, 2020.

[206] Z. Z. Li, X. Y. Zhu, Z. Lei, Z. X. Zhang. Deconfounding physical dynamics with global causal relation and confounder transmission for counterfactual prediction. In *Proceedings of the 36th AAAI Conference on Artificial Intelligence*, pp. 1536–1545, 2022. DOI: 10.1609/AAAI.V36I2.20044.

[207] S. C. Limeros, S. Majchrowska, J. Johnander, C. Petersson, D. F. Llorca. Towards explainable motion prediction using heterogeneous graph representations, [Online], Available: https://arxiv.org/abs/2212.03806, 2022.

[208] K. Fujii, K. Takeuchi, A. Kuribayashi, N. Takeishi, Y. Kawahara, K. Takeda. Estimating counterfactual treatment outcomes over time in complex multiagent scenarios, [Online], Available: https://arxiv.org/abs/2206.01900, 2022.

[209] J. Y. Su, S. Adams, P. A. Beling. Counterfactual multi-agent reinforcement learning with graph convolution communication, [Online], Available: https://arxiv.org/abs/2004.00470, 2020.

[210] N. Naderializadeh, M. Eisen, A. Ribeiro. Wireless power control via counterfactual optimization of graph neural networks. In *Proceedings of the 21st IEEE International Workshop on Signal Processing Advances in Wireless Communications*, Atlanta, USA, 2020. DOI: 10.1109/SPAWC48557.2020.9154336.

[211] H. R. Wu, W. Chen, S. Xu, B. Xu. Counterfactual supporting facts extraction for explainable medical record based diagnosis with graph network. In *Proceedings of Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 1942–1955, 2021. DOI: 10.18653/v1/2021.naacl-main.156.

[212] R. Xu, Y. Yu, C. Zhang, M. K. Ali, J. C. Ho, C. Yang. Counterfactual and factual reasoning over hypergraphs for interpretable clinical predictions on EHR. In *Proceedings of the 2nd Machine Learning for Health Symposium*, New Orleans, USA, pp. 259–278, 2022.

[213] B. L. Zhang, X. X. Guo, Q. F. Lin, H. R. Wang, S. B. Xu. Counterfactual inference graph network for disease prediction. *Knowledge-Based Systems*, vol. 255, Article number 109722, 2022. DOI: 10.1016/j.knosys.2022.109722.

[214] S. Y. Qin, S. L Jiang, J. P. Li, P. Balaprakash, R. C. Van Lehn, V. M. Zavala. Capturing molecular interactions in graph neural networks: A case study in multi-component phase equilibrium. *Digital Discovery*, vol. 2, no. 1, pp. 138–151, 2023. DOI: 10.1039/D2DD00045H.

[215] G. P. Wellawatte, A. Seshadri, A. D. White. Model agnostic generation of counterfactual explanations for molecules. *Chemical Science*, vol. 13, no. 13, pp. 3697–3705, 2022. DOI: 10.1039/D1SC05259D.

[216] J. R. Li, Y. Horiguchi, T. Sawaragi. Counterfactual inference to predict causal knowledge graph for relational transfer learning by assimilating expert knowledge -relational feature transfer learning algorithm. *Advanced Engineering Informatics*, vol. 51, Article number 101516, 2022. DOI: 10.1016/J.AEI.2021.101516.

[217] D. Pham, Y. F. Zhang. Counterfactual based reinforcement learning for graph neural networks. *Annals of Operations Research*, published online. DOI: 10.1007/s10479-022-04978-9.

[218] C. J. Xiao, X. Xu, Y. Lei, K. P. Zhang, S. Y. Liu, F. Zhou. Counterfactual graph learning for anomaly detection on attributed networks. *IEEE Transactions on Knowledge and Data Engineering*, vol. 35, no. 10, pp. 10540–10553, 2023. DOI: 10.1109/TKDE.2023.3250523.

[219] Y. T. Wei, W. Z. Shu, Z. T. Cheng, W. X. Tai, C. J. Xiao, T. Zhong. Counterfactual graph learning for anomaly detection with feature disentanglement and generation (student abstract). In *Proceedings of the 38th AAAI Conference on Artificial Intelligence*, Vancouver, Canada, pp. 23682–23683, 2024. DOI: 10.1609/aaai.v38i21.30524.

[220] M. Gulzar, Y. Muhammad, N. Muhammad. A survey on motion prediction of pedestrians and vehicles for autonomous driving. *IEEE Access*, vol. 9, pp. 137957–137969, 2021. DOI: 10.1109/ACCESS.2021.3118224.

[221] A. Dorri, S. S. Kanhere, R. Jurdak. Multi-agent systems: A survey. *IEEE Access*, vol. 6, pp. 28573–28593, 2018. DOI: 10.1109/ACCESS.2018.2831228.

[222] J. Foerster, G. Farquhar, T. Afouras, N. Nardelli, S. Whiteson. Counterfactual multi-agent policy gradients. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence*, New Orleans, USA, pp. 2974–2982, 2018. DOI: 10.1609/AAAI.V32I1.11794.

[223] S. C. Huang, A. Pareek, S. Seyyedi, I. Banerjee, M. P. Lungren. Fusion of medical imaging and electronic health records using deep learning: A systematic review and implementation guidelines. *npj Digital Medicine*, vol. 3, Article number 136, 2020. DOI: 10.1038/S41746-020-00341-z.

[224] A. Holzinger, B. Malle, A. Saranti, B. Pfeifer. Towards multi-modal causability with graph neural networks enabling information fusion for explainable AI. *Information Fusion*, vol. 71, pp. 28–37, 2021. DOI: 10.1016/J.INFFUS.2021.01.008.

[225] O. Wieder, S. Kohlbacher, M. Kuenemann, A. Garon, P. Ducrot, T. Seidel, T. Langer. A compact review of molecular property prediction with graph neural networks. *Drug Discovery Today: Technologies*, vol. 37, pp. 1–12, 2020. DOI: 10.1016/j.ddtec.2020.11.009.

[226] A. R. N. Aouichaoui, F. Fan, S. S. Mansouri, J. Abildskov, G. Sin. Combining group-contribution concept and graph neural networks toward interpretable molecular property models. *Journal of Chemical Information and Modeling*, vol. 63, no. 3, pp. 725–744, 2023. DOI: 10.1021/acs.jcim.2c01091.

[227] X. X. Ma, J. Wu, S. Xue, J. Yang, C. Zhou, Q. Z. Sheng, H. Xiong, L. Akoglu. A comprehensive survey on graph anomaly detection with deep learning. *IEEE Transactions on Knowledge and Data Engineering*, vol. 35, no. 12, pp. 12012–12038, 2023. DOI: 10.1109/TKDE.2021.3118815.

[228] C. Agarwal, O. Queen, H. Lakkaraju, M. Zitnik. Evaluating explainability for graph neural networks. *Scientific Data*, vol. 10, no. 1, Article number 144, 2023. DOI: 10.1038/s41597-023-01974-x.

[229] J. Deng, W. Dong, R. Socher, L. J. Li, K. Li, F.-F. Li. ImageNet: A large-scale hierarchical image database. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, Miami, USA, pp. 248–255, 2009. DOI: 10.1109/CVPR.2009.5206848.

[230] M. Westerlund. The emergence of deepfake technology: A review. *Technology Innovation Management Review*, vol. 9, no. 11, pp. 40–53, 2019. DOI: 10.22215/timreview/1282.

[231] T. Xiao, S. S. Liang, Z. Q. Meng. Hierarchical neural variational model for personalized sequential recommendation. In *Proceedings of World Wide Web Conference*, San Francisco, USA, pp. 3377–3383, 2019. DOI: 10.1145/3308558.3313603.

[232] T. Xiao, Z. Y. Chen, Z. M. Guo, Z. Y. Zhuang, S. H. Wang. Decoupled self-supervised learning for graphs. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, New Orleans, USA, 2022.

[233] B. Suh, L. C. Hong, P. Pirolli, E. H. Chi. Want to be retweeted? Large scale analytics on factors impacting retweet in twitter network. In *Proceedings of IEEE Second International Conference on Social Computing*, Minneapolis, USA, pp. 177–184, 2010. DOI: 10.1109/SOCIALCOM.2010.33.

[234] P. Mernyei, C. Cangea. Wiki-CS: A Wikipedia-based benchmark for graph neural networks, [Online], Available: https://arxiv.org/abs/2007.02901, 2020.

[235] W. Shalaby, W. Zadrozny, H. X. Jin. Beyond word embeddings: Learning entity and concept representations from large scale knowledge bases. *Information Retrieval Journal*, vol. 22, no. 6, pp. 525–542, 2019. DOI: 10.1007/S10791-018-9340-3.

[236] R. J. Wang, Y. C. Yan, J. L. Wang, Y. T. Jia, Y. Zhang, W. N. Zhang, X. B. Wang. AceKG: A large-scale knowledge graph for academic data mining. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, Torino, Italy, pp. 1487–1490, 2018. DOI: 10.1145/3269206.3269252.

[237] F. Frasca, E. Rossi, D. Eynard, B. Chamberlain, M. Bronstein, F. Monti. SIGN: Scalable inception graph neural networks, [Online], Available: https://arxiv.org/abs/2004.11198, 2020.

[238] S. M. Kazemi, R. Goel, K. Jain, I. Kobyzev, A. Sethi, P. Forsyth, P. Poupart. Representation learning for dynamic graphs: A survey. *Journal of Machine Learning Research*, vol. 21, no. 70, pp. 1–73, 2020.

[239] J. E. Perry-Smith, C. E. Shalley. The social side of creativity: A static and dynamic social network perspective. *The Academy of Management Review*, vol. 28, no. 1, pp. 89–106, 2003. DOI: 10.2307/30040691.

[240] B. Ran, D. Boyce. *Modeling Dynamic Transportation Networks: An Intelligent Transportation System Oriented Approach*, Berlin, Germany: Springer, 1996. DOI: 10.1007/978-3-642-80230-0.

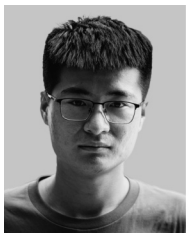
[241] T. Xiao, S. S. Liang, Z. Q. Meng. Dynamic collaborative recurrent learning. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, Beijing, China, pp. 1151–1160, 2019. DOI: 10.1145/3357384.3357901.

[242] T. Xiao, J. X. Ren, Z. Q. Meng, H. Sun, S. S. Liang. Dynamic Bayesian metric learning for personalized product search. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, Beijing, China, pp. 1693–1702, 2019. DOI: 10.1145/3357384.3358057.

[243] Y. N. You, T. L. Chen, Y. D. Sui, T. Chen, Z. Y. Wang, Y. Shen. Graph contrastive learning with augmentations. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, Vancouver, Canada, 2020.

[244] H. R. Zhang, Q. T. Wu, J. C. Yan, D. Wipf, P. S. Yu. From canonical correlation analysis to self-supervised graph neural networks. In *Proceedings of the 35th International Conference on Neural Information Processing Systems*, 2021.

[245] Y. X. Wang, M. I. Jordan. Desiderata for representation learning: A causal perspective, [Online], Available: https://arxiv.org/abs/2109.03795, 2021.



**Zhimeng Guo** received the B. Sc. degree in computer science from the University of Electronic Science and Technology of China (UESTC), China in 2021. He is currently a Ph. D. degree candidate in College of Information Sciences and Technology (IST), The Pennsylvania State University (PSU), State College, USA.

His research interests include robustness and alignment of foundation models.

E-mail: zzg5107@psu.edu

ORCID iD: 0000-0002-3921-7640

**Zongyu Wu** received the B. Eng. degree in computer science and technology from Chongqing University, China in 2023. Currently, he is a Ph. D. degree candidate in the Department of Information Sciences and Technology, The Pennsylvania State University, USA.

His research interests include trustworthy and effective foundation models.

E-mail: zongyuwu@psu.edu

**Teng Xiao** is a fourth-year Ph. D. student in the College of Information Sciences and Technology, The Pennsylvania State University, USA. He aims to develop Human-Centered AI using machine learning and reinforcement learning to enhance human decision-making and model trustworthiness in applications such as information and social systems, language model alignment, and scientific discovery. He has extensively published his work in top-tier conferences, including ICML, KDD, WWW, CIKM, AAAI, WSDM, NeurIPS.

His research interests include machine learning and reinforcement learning.

E-mail: tvx5054@psu.edu

**Charu Aggarwal** received the B. Sc. degree in computer science from the Indian Institute of Technology, Kanpur in 1993 and the Ph. D. degree in operations research (focus: mathematical optimization) from the Massachusetts Institute of Technology, USA in 1996. He is a distinguished research staff member (DRSM) at the IBM T. J. Watson Research Center in Yorktown Heights, USA. He has worked extensively in the field of data mining, with particular interests in data streams, privacy, uncertain data and social network analysis. He has authored 10 books, over 400 papers in refereed venues, and has applied for or been granted over 80 patents. His h-index is 136. He is a recipient of the IEEE ICDM Research Contributions Award (2015) and the ACM SIGKDD Innovation Award (2019), which are the two most prestigious awards for influential research in data mining. He is also a recipient of the W. Wallace McDowell Award, the highest award given by the IEEE Computer Society across the field of computer science. He also received the ACM SIGKDD Service Award for service contributions to the data mining community. He has served as the general or program co-chair of the IEEE Big Data Conference (2014), the ICDM Conference (2015), the ACM CIKM Conference (2015), and the KDD Conference (2016). He has served as the editor-in-chief of the *ACM SIGKDD Explorations* and is currently an editor-in-chief of the *ACM Transactions on Knowledge Discovery and Data Mining* as well as that of ACM Books. He is a fellow of the IEEE (2010), ACM (2013), and the SIAM (2015) for "contributions to knowledge discovery and data mining algorithms."

His research interests include applications of mathematical models to machine learning, data mining, and artificial intelligence, with particular interests in data streams, privacy, uncertain data, and social network analysis.

E-mail: charu@us.ibm.com

**Hui Liu** received the Ph. D. degree in electrical engineering from Southern Methodist University, USA in 2015. She is an assistant professor in the Computer Science and Engineering Department, Michigan State University, USA.

Her research interests include trustworthy AI, designing data mining algorithms for wireless communication of smart devices, and applying machine learning and data mining in wireless communications.

E-mail: liuhui7@msu.edu

**Suhang Wang** received the B. Sc. degree in electrical and computer engineering from Shanghai Jiao Tong University, China in 2012, the M. Sc. degree in electrical engineering from University of Michigan, USA in 2013, and the Ph. D. degree in computer science from Arizona State University, USA in 2018. He is an associate professor of the College of Information Sciences and Technology, The Pennsylvania State University, USA. He is an associate editor for several journals and serves as regular journal reviewers and numerous conference program committees. He has published over 140 innovative works in highly ranked journals and top conference proceedings, which have received extensive coverage in the media.

His research interests include graph mining, data mining and machine learning.

E-mail: szw494@psu.edu (Corresponding author)
ORCID iD: 0000-0003-3448-4878