A SURVEY ON GRAPH COUNTERFACTUAL EXPLANATIONS: DEFINITIONS, METHODS, EVALUATION

A PREPRINT

Mario Alfonso Prado-Romero
Gran Sasso Science Institute
L'Aquila, Italy
marioalfonso.prado@gssi.it

Bardh Prenkaj
Sapienza University of Rome
Rome, Italy
prenkaj@di.uniroma1.it

Giovanni Stilo
University of L'Aquila
L'Aquila, Italy
giovanni.stilo@univaq.it

Fosca Giannotti
Scuola Normale Superiore
Pisa, Italy
fosca.giannotti@sns.it

October 24, 2022

ABSTRACT

In recent years, Graph Neural Networks have reported outstanding performance in tasks like community detection, molecule classification and link prediction. However, the black-box nature of these models prevents their application in domains like health and finance, where understanding the models' decisions is essential. Counterfactual Explanations (CE) provide these understandings through examples. Moreover, the literature on CE is flourishing with novel explanation methods which are tailored to graph learning.

In this survey, we analyse the existing Graph Counterfactual Explanation methods, by providing the reader with an organisation of the literature according to a uniform formal notation for definitions, datasets, and metrics, thus, simplifying potential comparisons w.r.t to the method advantages and disadvantages. We discussed seven methods and sixteen synthetic and real datasets providing details on the possible generation strategies. We highlight the most common evaluation strategies and formalise nine of the metrics used in the literature. We first introduce the evaluation framework GRETEL and how it is possible to extend and use it while providing a further dimension of comparison encompassing reproducibility aspects. Finally, we provide a discussion on how counterfactual explanation interplays with privacy and fairness, before delving into open challenges and future works.

Keywords Explainability, Explainable AI, Machine Learning, Post-hoc Explanation, Graphs, Graph Neural Networks

1 Introduction

Artificial Intelligence has received an enormous amount of contributions in the last decade. In particular, deep neural networks have been adopted massively in domains among which computer vision [53, 117, 118], natural language processing [16, 33, 127], recommender systems [56, 78, 4], and anomaly detection [79, 96, 112]. Recently, Graph Neural Networks - hereafter GNNs - have overcome a plethora of challenges in many graph mining tasks that encompass vertex classification [50, 97, 109, 139, 162, 160, 165], link predictions [55, 59, 97, 142, 151, 157], community detection [20, 22, 122, 155], and graph classification [10, 69, 146, 151]. Intelligent systems can be used in decision support scenarios where processing requests becomes cumbersome and time-consuming for humans [42]. For example, hospitals might employ an automatic system that aids the healthcare personnel in deciding whether a patient is susceptible to heart-related diseases or not, considering current medical conditions and past disease history. Banks might have a system that decides whether clients have their loans approved. Social networks could employ an intelligent strategy to detect users that violate their terms of services and, consequently, ban them from the platform. Lastly, pharmaceutical companies can use these kind of decision support systems (DSS) for drug repurpusing.

Although intelligent systems have reached outstanding performances[1], the widely used deep neural networks suffer from the so-called *black-box* problem [5, 101]. Black-box models hinder the understanding of the decision process adopted when forecasting outcomes [39]. Neural networks exploit non-reversible and nonlinear activation functions from one layer to the next to learn feature representations in nonlinear vector spaces [23, 34] becoming opaque to the users and belonging to the class of black-box models. The opposite of a black box is a system made up of inner processes that we can easily inspect. These models are often referred to as white-boxes or *transparent* models [74]. Generally, black- and white-box models are at the extremes of the spectrum in Data Science (see Figure 1). While

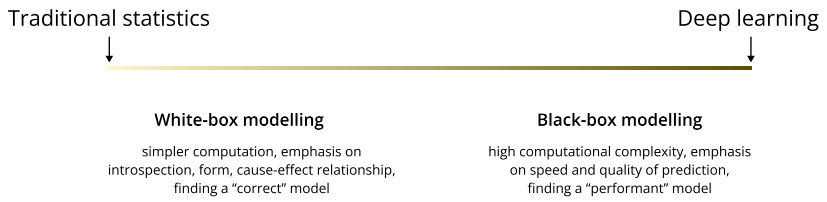

Traditional statistics
Deep learning
White-box modelling
simpler computation, emphasis on introspection, form, cause-effect relationship, finding a "correct" model
Black-box modelling
high computational complexity, emphasis on speed and quality of prediction, finding a "performant" model

Figure 1: Interpretability (left) vs performance (right) spectrum in Data Science.

white-box models are preferred when making critical decisions that might change the course of events in a business [32, 129, 130], black-box models are more performant and can generalise even with high-dimensional input features in scenarios such as predicting protein relations [3, 81, 140, 159], student dropout prediction [15, 105, 104, 135], and trend prediction [7, 70, 131, 148].

As mentioned above, GNNs have reached outstanding performances in several graph mining tasks. GNNs take as input a graph structure which contains vertices representing concepts and edges depicting their relationships. Examples of graphs are Facebook's user friendship network and the PPI which collect proteins and their relationships. In other words, vertices are users (or proteins) and the edges are the friendships (or chemical relationships) among them. One interesting prediction task in the Facebook's graph is forecasting whether two users will become friends in the near future: i.e. link prediction[2]. In this way, the model might discover a relationship between two users that have similar interests and help long-lost friends rekindle their childhood relationship[3]. Specifically, a GNN accepts a graph, with attributes associated to its vertices and edges, and produces a transformed graph that has learnt latent features for all these attributes. The latent features can then be used to perform predictions. Notice that the connectivity patterns induced from the edges of the graph are used to enforce relationships in the latent vector space of the features learnt through the graph transformation (see Section 2.1 for further details).

The ability to explain the predictions is essential to allow users and service providers [85, 46] make trusted decisions in critical domains. For this reason, nowadays, deep models are not yet widespread in production in critical domains like health and finance due to legal restrictions [98]. The literature has received many contributions in interpreting black-box models via feature-based analysis [45] and counterfactual explanations [29, 44, 63, 66, 75, 137]. Models that provide explanations over the predictions made by black-box models through the analysis of the features that contributed to the outcome are denoted as feature-based explainers. However, explanations that reveal the *why* of a prediction are not sufficient to understand how to change the outcome of a specific model. For this reason, explainers that provide examples of what input features to change to predict a different outcome are necessary to describe cause-effect relationships between the input data and the outcome [21, 85]. In other words, counterfactual explainability suggest which differences should the input have to induce a change in the model prediction. The mechanism that provides this kind of explainability are called *counterfactual* explainers.

According to the previous three example scenarios, counterfactual explainability could help, for example, obtain the following suggestions:

[1]AI has surpassed human gold standards in object recognition [123, 164] and playing games [17, 114, 115, 116] among some of the fields. We invite the reader to read Microsoft's Azure AI milestones at `https://www.microsoft.com/en-us/research/blog/azure-ai-milestone-microsoft-kear-surpasses-human-performance-on-commonsenseqa-benchmark/`, and AlphaGo's triump over world-class Go player, Lee Se-dol, at `https://www.bbc.com/news/technology-35785875`.

[2]Facebook has a friend suggestion functionality that takes information from already-established friendships among two users, and suggest them other similar profiles to make plausible new friendship connections. Because Facebook's prediction algorithms remain confidential and proprietary, we can assume that GNNs would be beneficial in this particular task due to their intrinsic characteristics of extrapolating information from the neighbourhoods of the graph vertices (see Section 2.1)

[3]`https://www.rewire.org/reconnecting-with-old-friends/`

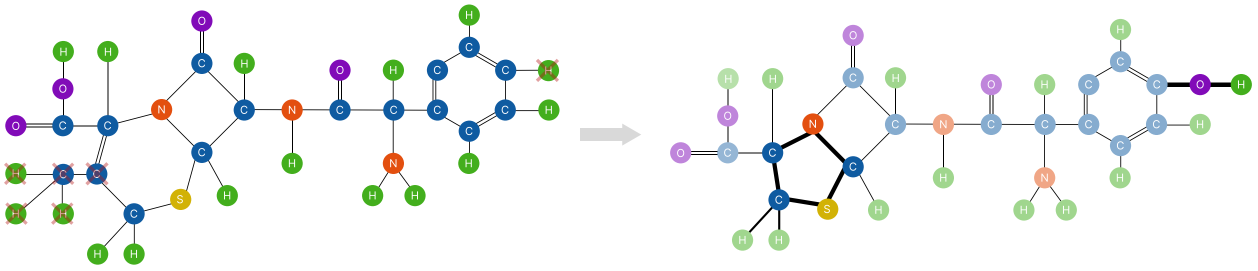


Figure 2: Drug repurpusing mechanism. Passing from cephallexin (left) to amoxicillin (right) by removing atoms and reforming hexagonal connections into pentagons. Here, the counterfactual explanation consists of the highlighted edges and vertices.

- *Heart disease Clinical DSS (CDSS)* - Let us assume that a patient with a body mass index (BMI) of 31.4 (current medical conditions) is the input of the CDSS. A counterfactual explanation of this instance could be *if the patient reduced her BMI by 20%, then she would drastically reduce the chances of getting a heart disease or complications thereof in the near future.* This kind of explanation might provide the healthcare personnel with the right input towards a personalised weight loss plan for the patient.
- *Loan DSS* - In this case, a bank customer with a \$15,000 deposit and a credit score of 25 asks for a loan of \$150,000. Let us suppose, that the DSS employed refuses the requests and provides a counterfactual explanation *if the customer*[4] *increased her credit score by 2% and her bank deposit up to \$25,000, then her loan would be granted.* This particular counterfactual example gives direct input to the customer of how to act for her to be eligible for a loan approval. In this sense, counterfactual explainability provides exact steps towards controllable actions: i.e. the above customer needs to have an additional \$10,000 and in doing so she should pay her credit card debt such that her credit reaches 25.5 points.
- *User banning DSS* - Let us assume that a specific user has written a post that illegally tries to sell drugs in the vending section[5] of our bogus social network. In today's social networks, selling drugs online is a direct infringement of the terms of use and leads to apprehension besides getting the profile banned. Therefore, our DSS blocks the user's account indefinitely providing a counterfactual explanation as *if the user had not written the post about selling drugs, her account would have not been banned.* Explanations about violation of social network regulations can help auditors categorise banned users according to the severity of the violation, and provide a safer cyberspace for future users in the social network.
- *Drug repurpusing DSS* - Let us assume that a pharmaceutical laboratory is about to discover a drug that is based on a first generation cephalosporins - i.e. cephalexin[6] - to help cure bacterial infection complications such as tonsillitis, bronchitis, sinusitis, and dental abscess. When treating such diseases with cephalexin, our DSS gives a negative result providing a counterfactual explanation as *if we modified cephalexin's molecular structure as in Figure 2, then we would be able to treat the diseases.* This explanation is quite useful because it would give sprout to the aminopenicillins class of antibiotics, in particular to the drug corresponding to amoxicillin known for its faster treatments and lower risk of side effects.

In the scenarios explained above, counterfactual explainability provides more detailed investigation w.r.t. feature-based explainability. Counterfactuals allow black-box models have a breakthrough in critical domains by building up trust, being transparent, and providing explicit feedback to non experts. Additionally, counterfactual systems are useful to reveal intrinsic biases in the prediction model. In particular, consider the example of the CDSS deciding on heart disease complications for a patient. Imagine, now, that having predispositions towards heart diseases might reject a life-saving surgery of a certain patient. Hence, if the CDSS decides that the patient might suffer from heart diseases, her surgery gets rejected. While patients can control their BMI and decrease to a normal interval such that the CDSS gets her clear for heart-related conditions, the CDSS might decide that heart diseases are related to race [94, 143]. In these cases, race is an uncontrollable variable that undermines the fairness, a pillar requisite in today's democratic countries, of predictive models [24, 18].

[4]We invite the reader to read this paragraph as a mere example without any formal financial background to it.

[5]Illicit activities have occurred throughout the years in world-known social networks: e.g. `https://www.washingtonpost.com/technology/2020/05/27/facebook-sec-whistleblower/`, `https://foreignpolicy.com/2020/12/15/latin-american-drug-cartels-instagram-facebook-tiktok-social-media-crime/`, `https://www.thesun.co.uk/news/5330309/inside-facebook-group-sell-drugs-openly-shut-down/`.

[6]`https://pubchem.ncbi.nlm.nih.gov/compound/cephalexin`

Thus, it is important that DSS do not perpetuate and exacerbate existing biases and inequalities embedded, for instance, in healthcare tools, therapeutics and systems that see certain groups receiving substandard or no care at all [95]. Counterfactual explanations, as mentioned before, can aid in identifying cases where bias arises and help address challenges relying to ethical principles rather than performance-driven concepts. Yet, eradicating bias from data is a completely different challenge which is out of scope for this survey.

Finally, from the market point-of-view, the mean cost of developing a new drug ranges from \$314 million to \$2.8 billion as of 2018 [144]. Additionally, a drug's approval rate entering clinical development is a mere 12%, which gives a clear indication that the majority of the money spent are wasted. At the same time, the average amount of completing a full cycle of clinical trials for a drug has decreased, meaning that pharmaceutical companies need to re-iterate their tests before monetising their drug. The factors that play an important role in this increase in prices include, but are not limited to, the increased clinical trial complexity, larger trial sizes, changes in protocol design to gather health technology assessment information, and testing on comparator drugs to alleviate customer demands[7]. Having clear counterfactual methods to interpret the causes of the outcome, in application scenarios such as drug re-purposing, gives large possibilities to reduce costs and increase approval rates. Hence, pharmaceutical companies are interested to adopt novel explainable graph learning approaches to understand the trial scenarios without delving into costly tests.

For the aforementioned reasons, in this survey, we explore counterfactual explanations on geometric deep learning which is an umbrella term for emerging techniques attempting to generalise deep neural models to non-Euclidean domains. Here, we focus on graph data and the prediction of them via GNNs. To the best of our knowledge, this is the first survey that treats counterfactual explainability on graphs.

For completeness purposes, in Section 1.1 we discuss several surveys [9, 44, 121, 133] which tackle the interpretation of black-boxed deep neural networks but that are not specifically focused on GNN and counterfactual explanation at the same time. Finally, in Section 1.2 we conclude the introduction of this survey by highlighting its contributions.

1.1 Differences between this survey and former ones

Graph Counterfactual Explainability (GCE) is still in its early stages since significant attention towards graph learning has risen sharply in the last decade. To the best of our knowledge, this is the first extensive study that summarises and classifies the works proposed in this particular domain. Nevertheless, for completeness purposes, we include surveys on graph factual explainability [6, 153] and counterfactual explainability on other data structures [9, 44, 133]

We analysed the existing surveys according to seven dimensions, reflecting the scope, the domains and typical workflow of GCE's research area: *Domain* considers if the survey analyses the problem and the domain of the surveyed studies, the *Prediction Task* if the survey analyses the goal of the explainable method, *Definition* if the survey highlights the formal definition adopted by the considered explainable methods, *Explainer Adaptation* that depict whether the survey discusses on the explainers mechanisms by considering those that encompass or extends a base explainer, if the survey analyses the *Metrics* of evaluations, and if the survey discussed any *Evaluation Framework*, and if the survey discusses on aspects related to *Privacy Infringement Mitigation*. As argued in this Section, we believe that these dimensions provide a better framework to categorise and compare existing surveys. In the following, we describe each dimension in more detail and we show, as summarised in Table 1 ($\times$ depicts a missing aspect; ✓depicts a covered aspect; $\sim$ depicts a partially covered aspect).

1. **Domain**: It represents the different types of the structures of graphs taken into consideration to tackle the GCE problem. We identified four main categories of graph data: i.e. synthetic, social, molecular, and -omics networks. Other surveys concentrate on synthetic, social [6, 153] and molecular networks [153] because they represent traditional structures in graph deep learning. Besides extensively discussing the first three graph data types, we also shed light on the structure of -omics networks [80] (e.g. protein-protein interactions) which are novel representations that combine network science and biology to analyse the interconnection of biological processes. With the introduction of -omics networks, we aim to provide detail on the support of GCE in network medicine to understand biological mechanisms and address both diagnostics and therapeutic aspects therein.
2. **Prediction Task**: It indicates the types of tasks applied to each method in this survey. We have identified three types of tasks in the works in GCE: i.e. vertex and graph classification, and link (edge) prediction. There is a substantial difference between the three prediction tasks mentioned above since they rely on different concepts. Node classification engages in predicting the identity/role of each vertex v in a graph by analysing the role of the nodes in v's neighbourhood. A typical vertex classification example is Zachary's karate club [156], a social

[7]`https://www.policymed.com/2014/12/a-tough-road-cost-to-develop-one-new-drug-is-26-billion-approval-rate-for-dr`
`html`

Table 1: Qualitative comparison with other surveys in the literature according to the coverage of relevant dimensions reflecting the scope and workflow of GCE approaches. $\times$ depicts a missing aspect; ✓depicts a covered aspect; $\sim$ depicts a partially covered aspect. For completeness purposes, we include surveys on graph factual explainability [6, 153] and counterfactual explainability on other data structures [9, 44, 133].

			Surveys on Explainability					
			Considering graphs data			Considering other data types		
			Counterfactual	Factual		Counterfactual		
			[this]	Amara et al. [6]	Yuan et al. [153]	Artlet and Hammer [9]	Guidotti [44]	Verma et al. [133]
Domain	Synthetic networks		✓	✓	✓	×	×	×
	Social networks		✓	✓	✓	×	×	×
	Molecular networks		✓	×	✓	×	×	×
	Omics Networks		✓	×	×	×	×	×
Prediction Task	Vertex classification		✓	×	✓	×	×	×
	Link (edge) prediction		✓	×	×	×	×	×
	Graph classification		✓	×	✓	×	×	×
Definition			✓	×	×	✓	✓	✓
Explainer Adaptation			✓	×	✓	✓	×	×
Metrics	Runtime		✓	✓	×	×	✓	✓
	Calls to Oracle		✓	×	×	×	×	×
	Correctness		✓	×	×	×	×	×
	Graph Edit Distance		✓	×	×	×	×	×
	Sparsity		✓	✓	✓	×	×	✓
	Oracle Accuracy		✓	✓	✓	×	×	×
	Fidelity		✓	✓	✓	×	×	×
Evaluation Framework	Datasets	Synthetic	✓	✓	✓	×	×	×
		Real	✓	✓	✓	×	✓	✓
	Reproducibility		✓	∼	∼	×	∼	×
	Extensibility		✓	×	×	×	×	×
	Supported Methods		✓	✓	✓	×	✓	×
Privacy, Fairness, and Trustworthiness			✓	×	×	×	×	×

network comprising individuals that belong to one of the two available karate clubs (i.e. bipartite graph). The vertices represent individual karate practitioners and the edges are interaction between members of the club outside the dojo. The prediction, in this scenario, is to classify whether a given member will change their[8] club. Link prediction tasks are useful when we want to predict relationships between vertices. For example, given some vertices that represent entities in an image, the goal is to predict which of the vertices shares an edge. Lastly, in graph classification, the goal is to predict the property of an entire graph. For example, we want to predict whether a molecule - represented as a graph - binds to a receptor implicated in a specific disease.

3. **Definition**: While graph neural networks [71, 113, 128, 54] and counterfactual explainability [9, 44, 133] have been extensively defined, to the best of our knowledge, this survey is the first contribution in the literature with a uniform GCE definition. Our goal is to provide a formalisation of the GCE problem and adapt what has been already proposed in the literature. Thus, future researchers can use the formalisation provided to align their work with the state-of-the-art.

4. **Explainer Adaptation**: As done in surveys for counterfactual explainability on non-graph structures [153], some methods in GCE use factual explainability as a starting point and change the features of the factual to produce the counterfactual. Here, we analyse the counterfactual explanation technique as an adaptation similar to what has been done in the factual explanation approaches or as an extension of image-based explainability.

5. **Metrics**: To correctly evaluate the performances of a given explainer, several metrics are commonly used, such as the oracle accuracy, sparsity, and accuracy. Amara et al. [6] covers also the runtime of the explanation technique besides the three abovementioned metrics covered in [153]. The counterfactual surveys in non-graph data [46, 133] also consider the runtime as an evaluation metric. However, to distinguish counterfactual explainability on graphs from other categories of explainability, we propose the graph edit distance and the correctness of the explainer as useful metrics for future research.

6. **Evaluation Framework**: It is crucial to provide a uniform benchmarking system for evaluating the methods w.r.t. standardised metrics and datasets in the literature. While the other surveys [6, 133, 153] enlist the characteristics and pitfalls of synthetic/real datasets, only Guidotti [44] and Yuan et al. [153] delve further into detail to explain the reproducibility[9] of their evaluation framework by providing links to repositories containing the code base. These publicly accessible projects have a handful of explanation methods of the literature implemented and ready for usage. Nevertheless, these evaluation frameworks [6, 44, 153] are difficult to extend and reuse for experiments in scenarios different from those already included. In this survey, we discuss GCE evaluation tools posing the attention on GRETEL which incorporates state-of-the-art explainers,

[8]Throughout this survey, we use the *singular they* when the gender of the subject is unknown. `https://www.merriam-webster.com/words-at-play/singular-nonbinary-they`

[9]Notice that Amara et al. [6] briefly discuss their explainability but do not provide evidence on what parameters to provide in the code base for reproduction purposes.

datasets and metrics. The framework allows researchers interested in GCE to extend abstract classes of counterfactual explainers and use them for specific prediction tasks. Finally, we note GRETEL's advantage under the reproducibility aspect.

7. **Privacy, Fairness, and Trustworthiness**: Infringement of privacy is a complex issue involving diverse meanings in different cultures. While not every intrusion violates privacy, it is difficult to delineate what a violation is at any given moment because of shifting social expectations and the increasing interaction between people of different cultures. While this survey is limited in treating the infringement of privacy in graph-like data and not human privacy per se, preserving privacy is of paramount importance, for example, in social networks. The enlisted surveys in (counter)factual explainability do not shed light on the possible privacy infringements of sensitive data. Furthermore, the counterfactual explainer methods proposed in the literature are not inherently built to preserve privacy. To this extent, we show how to construct privacy-preserving explanation strategies. Finally, we discuss how counterfactual explainability can be harnessed to tackle the inherent unfairness and trustworthiness of the underlying prediction model. In this way, a counterfactual example generated from a biased predictor can be used as an intervention mechanism to cope with such situations.

As highlighted in Table 1, the main limitations of factual surveys on graph data are the following:

1. The surveys do not consider diverse graph data but limit themselves to synthetic, social, and molecular networks.
2. The surveys do not explain the characteristics of link prediction tasks and their significant differences from graph and vertex classification.
3. The surveys summarise the types of explainers by simply enumerating them without providing a critical analysis of the main limitations. Some explainers are suitable for all prediction tasks; others cope with some of them only. This discussion is absent in the other surveys.
4. The surveys fail to thoroughly describe reproducibility and extensibility aspects of their evaluation frameworks. Additionally, these frameworks do not allow to attach new explainer modules and test them in new experimental scenarios.
5. The surveys avoid discussing privacy infringement that the explainers could introduce when providing factual examples to the end-user. We believe this aspect is essential, considering the wide usage of social networks and the information generated therein.
6. The surveys lack treating the fairness and trustworthiness of explainers and the prediction model they are built upon. We believe that exposing bias from prediction methods via counterfactual explainability is a task that will grasp the attention of researchers in this area in the near future according to recent AI regulations worldwide.

1.2 Contributions of this survey

To the best of our knowledge, this survey is the first in the literature that treats GCE. As detailed in the previous Section, our work transcends the mere feature-based (factual) explainability in graphs and provides an in-depth discussion and more systematic review that the other surveys available in the literature. In summary, the main contributions of this survey are the following:

1. We organise the existing counterfactual example definitions in the literature according to a uniform formal notation, thus facilitating a straightforward comparison w.r.t. to their strengths and pitfalls.
2. To the best of our knowledge, we are the first to propose a formalisation of GCE according to a multi-class prediction problem. We also derive the global minimal counterfactual example on a particular graph according to a black-box prediction model.
3. We propose a classification of the existing methods in the literature according to several dimensions which help the reader to easily adopt alternative methodologies that better suit their scenario.
4. We provide the reader with a summary of strengths and weaknesses of the counterfactual explainer methods.
5. We discuss extensively the benefits of the evaluation protocol enlisting the synthetic and real datasets used in the literature, and the supported methods. To this end, we describe GRETEL, a fully-extensible and reproducible GCE evaluation framework. We argue that GRETEL is the first framework in the literature that concentrates in providing a highly modular architecture permitting future researcher to plug-and-play bespoke explainer models.

6. We consider a fundamental yet absent aspect in the literature - i.e. privacy, fairness, and trustworthiness - necessary to comply with AI regulations being adopted worldwide.
7. Finally, we provide the reader with insights on the open challenges - e.g. counterfactuality in multi-label classification - in this research field to be tackled in the near future.

We organise the rest of this survey as follows. Section 2 describes the relevant notation and background concepts in the GCE domain. Here, we propose a definition of graph counterfactual examples in a multi-class prediction scenario (see Section 2.2). In Section 3, we discuss the methods present in the literature (see Section 3.3) by providing an extended classification which comprises eight dimensions (see Section 3.2). Section 4 is dedicated to the evaluation protocol of the GCE methods providing a discussion of the datasets (see Section 4.1), the measures (see Section 4.2) and evaluation strategies (see Section 4.3) adopted in the literature. In Section 5, we discuss the evaluation tools specifically designed for GCE methods by posing the attention on GRETEL for which we briefly report its design principles and core components (see Section 5.1). In Section 5.2, we argue about GRETEL's intrinsic reproducibility and extensibility features as a supportive mechanism for future plug-and-play usage in the academia. Finally, Section 7 summarises open issues and promising directions in GCE, and Section 8 presents the concluding remarks.

2 A formal representation of the GCE problem

Before detailing a formal definition for a counterfactual explanation in graphs, we provide the reader with the necessary building blocks and background concepts (see Section 2.1). Additionally, we uniform the various counterfactual definitions provided in the literature according to a formal notation. Thus, we aim to provide future researchers with insight about the strengths, weaknesses, and assumptions of each of the enlisted definitions (see Sections 2.2.1 and 2.2.2). Finally, to contribute to the literature of GCE, we provide a formal definition for graph counterfactual examples in a multi-class prediction scenario (see Section 2.2.3).

2.1 Background concepts

Here we introduce definitions of the relevant entities that constitute the background on graph counterfactual explainability. We invite the reader to use Table 2 as a quick reference for the formal notation used throughout this survey.

Table 2: Formal notation used in this survey.

Notation	Meaning
$G = (V, E)$	A graph with the vertex set V and edge set E
v_i	A vertex $v_i \in V$
$e = (v_i, v_j)$	An edge $e \in E$ with its incident vertices v_i and v_j
$a(v_i)$	The attributes of vertex v_i
$\bar{a}(e)$	The attributes of edge e
$\tilde{a}(G)$	The attributes of graph G - dubbed global context
$\mathcal{G}$	The dataset comprising of a set of graphs $G_1, \ldots, G_{\lvert\mathcal{G}\rvert}$
$adj(v_i)$	Set of adjacent vertices of v_i
$co(v_i)$	Set of edges with incident vertex v_i
$d(v_i, v_j)$	Distance of the shortest path between v_i and v_j
A	The adjacency matrix of graph G
D	The degree matrix of graph G
F	The attribute matrix of the vertices of graph G
x	An instance to be examined
x'	A counterfactual instance
$\mathcal{F}_x$	The set of features for instance x
X	A set of original instances $\{x_1, \ldots, x_{\lvert X\rvert}\}$
X'	A set of counterfactual instances $\{x'_1, \ldots, x'_{\lvert X'\rvert}\}$
C	A set of counterfactual examples
Φ	A black-box model/Oracle
$\hat{y} = \Phi(x)$	The decision $\hat{y}$ of the model on x
$f_k(x, \Phi, X)$	A counterfactual explainer producing at most k examples

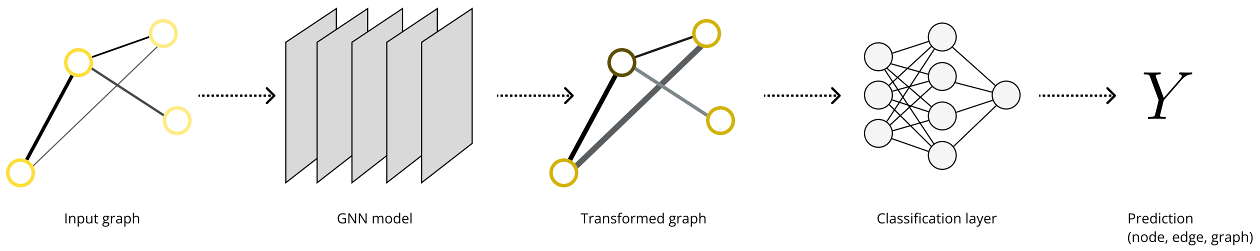


Figure 3: End-to-end prediction task of a GNN model.

2.1.1 What is a graph?

A graph $G = (V, E)$ is a data structure comprising of a set of vertices $V = \{v_1, ..., v_n\}$ and a set of edges $E = \{(v_i, v_j) \mid v_i \wedge v_j \in V\}$. In other words, a vertex $v \in V$ depicts an object/concept, whereas an edge $e = (v_i, v_j) \in E$ depicts the relationship between its two incident vertices $v_i, v_j \in V$. Graphs can be either directed or undirected. A directed graph has an asymmetrical edge set, whereas, an undirected graph depicts symmetrical relationships. In the undirected graph, $(v_i, v_j) = (v_j, v_i) \in E$. The neighbourhood of v_i, denoted by set $adj(v_i)$, contains all the vertices that are connect by an edge to v_i formally $adj(v_i) = \{v_j \in V \mid (v_i, v_j) \in E \vee (v_j, v_i) \in E\}$. Similarly, let $co(v_i)$ be the set of edges having v_i as an incident vertex, $co(v_i) = \{(v_i, v_j) \in E\} \cup \{(v_j, v_i) \in E\}$. Notice that vertices and edges can have categorical and numerical attributes. We denote with $a(v)$ the set of the values of the attributes of vertex $v \in V$, and with $\bar{a}(e)$ the set of values of the attributes of edge $e \in E$. Additionally, we depict with $\tilde{a}(G)$ the attributes of the entire graph G, also referred as global context. For convenience purposes, we define the adjacency matrix A, the degree matrix D, and the vertex attribute matrix F for graph G as follows:

$$A[v_i, v_j] = \begin{cases} 1 & \text{if } v_j \in adj(v_i) \\ 0 & \text{otherwise} \end{cases} \qquad D[v_i, v_j] = \begin{cases} |adj(v_i)| & \text{if } v_i = v_j \\ 0 & \text{otherwise} \end{cases} \qquad F[v_i, k] = a(v_i)[k]$$

2.1.2 Graph Neural Networks

Having formally defined the characteristics of a graph, we have the necessary tools to introduce graph neural networks (GNNs). In general, a neural network (NN) comprises of many interconnected layers of neurons that, upon receiving a firing signal, propagate information to the successive layers. We refer the reader to [38] for the formalisation of multiple-layered NNs. Similarly, a GNN is an optimisable transformation on all attributes (i.e. on nodes, edges, and global context) of a certain graph G which typically maintains its proximity characteristics: i.e. two nodes v_i and v_j that have similar attributes and neighbourhoods are near each other in the target embedding space.

Figure 3 illustrates the end-to-end prediction task of a GNN model. Generally, a GNN accept a graph G in input with attributes associated to its vertices, edges, and a global context associated with G. Then, the model produces a transformed graph which gets fed to a classification layer - e.g. a deep neural network - producing a prediction $\hat{y}$. The simplest GNN learns a new embedding for all attributes while not exploiting G's connectivity patterns. This kind of GNN uses a separate multilayer perceptron (MLP) on each component of the graph. For each node, the model applies an MLP and gets back an embedded set of attributes for each vertex. The same reasoning extends to edges and the global context. Recall that this simple GNN does not modify the structure of the graph. However, the transformed graph, shown in Figure 3, contains embedded attributes for each of its components. To make predictions on the transformed graph, we can collect information via pooling and proceed as follows:

- For each item to be pooled, we gather its embedding.
- We aggregate the gathered embeddings via a permutation-invariant operation (e.g. summation).

The pooling operation's functionality depends from the task that we want to perform (i.e. vertex, edge, graph classification).

To exploit the connectivity patterns in a graph G, we rely on three different concepts: i.e. random walks[10], message passing, and graph convolutional networks (GCNs). For completeness purposes, we provide a description for each of them and detail their similarities and differences.

Random Walks are stochastic processes that delineate a path that consists of a succession of random steps on a particular mathematical space [51]. Generally, a random walk process in graphs consists of the following three steps:

[10]Random walks can be considered as probabilistic graph visiting algorithms where each edge has a transition probability.

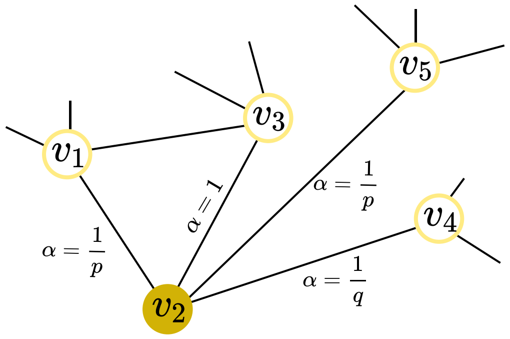


Figure 4: Illustration of a random walk procedure according to Node2Vec [43]. The walk transitioned from v_1 to v_2 and it needs to evaluate its next transition towards the vertices in $adj(v_2)$. The labels in the edges indicate the search biases α.

1. Given a graph $G = (V, E)$ and a starting vertex v_i, called the seed, select neighbour $v_j \in adj(v_i)$ with probability $\frac{1}{|adj(v_i)|} = \frac{1}{D[v_i,v_i]}$.
2. Move to the selected neighbour v_j, and repeat the same process as in the previous step until convergence.
3. The random sequence of vertices visited composes the random walk throughout the graph G.

One of the most influential works in vertex embeddings based on random walks is Node2Vec [43]. According to the original proposal, Node2Vec builds on top of a biased random walk process that efficiently exploits neighbourhood exploration in a breadth-first and depth-first search jointly. Given a source vertex v_i, the authors simulate a random walk of a specific length ℓ. The vertices in the walk are visited according to the following probability distribution:

$$P(v_j|v_i) = \begin{cases} \frac{\pi_{v_i,v_j}}{Z} & \text{if } v_j \in adj(v_i) \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

where π_{v_i,v_j} is the non normalised transition probability between vertices v_i and v_j, and Z is the normalising constant. To make the stochastic graph visiting described in Equation 1, Node2Vec adopts a second order random walk with two parameters p and q. Figure 4 illustrates a small view of a random walk in Node2Vec. Suppose that the walk just traversed the edge (v_1, v_2) and is now deciding what path to traverse out of v_2 according to the probabilities $\pi_{v_2,v_j}\ \forall v_j \in adj(v_2)$. Node2Vec sets the non normalised transition probability from any vertex v_i to v_j as $\pi_{v_i,v_j} = \alpha_{p,q}(v_k, v_j) \cdot w(v_i, v_j)$ such that:

$$\alpha_{p,q}(v_k, v_j) = \begin{cases} \frac{1}{p} & \text{if } d(v_k, v_j) = 0 \\ 1 & \text{if } d(v_k, v_j) = 1 \\ \frac{1}{q} & \text{if } d(v_k, v_j) = 2 \end{cases} \quad (2)$$

where v_k is the vertex visited immediately before v_i, $d(v_i, v_j)$ is the distance of the shortest path between v_i and v_j, and $w(v_i, v_j)$ is the weight of the edge $e = (v_i, v_j)$. Parameters p and q control the velocity of exploring and leaving the neighbourhood of a starting vertex v_i. For a detailed discussion about the trade-off between p and q, we refer the reader to the original paper [43]. Besides being able to generate vertex embeddings, Node2Vec can be exploited to learn edge embeddings. Given two vertices v_i and v_j, Node2Vec relies on a binary operator over the attributes $a(v_i)$ and $a(v_j)$ to generate an embedded representation $g(v_i, v_j)$ for the edge[11] $e = (v_i, v_j)$. Other random-walk-based methods are LINE [125], DeepWalk [99], RW^2 [81], HeteSpaceyWalk [48], and GloVeNoR [65].

Message passing: assumes that neighbouring vertices or edges exchange information and influence each other's updated embeddings. Without loss of generality, message passing works in three steps:

- For each vertex $v_i \in V$, gather the vertex embeddings of $adj(v_i)$ according to a function g.
- Aggregate the embeddings (messages) via an aggregation function f (e.g. summation).
- All pooled messages are passed through an update function (e.g. a trained NN).

Similar to the application of the pooling operation, message passing can occur between either vertices or edges. Figure 5 depicts a similar operation w.r.t. standard convolution. In other words, message passing and convolutions are

[11]Notice that Node2Vec can also generate embeddings for non existing edges in the original graph.

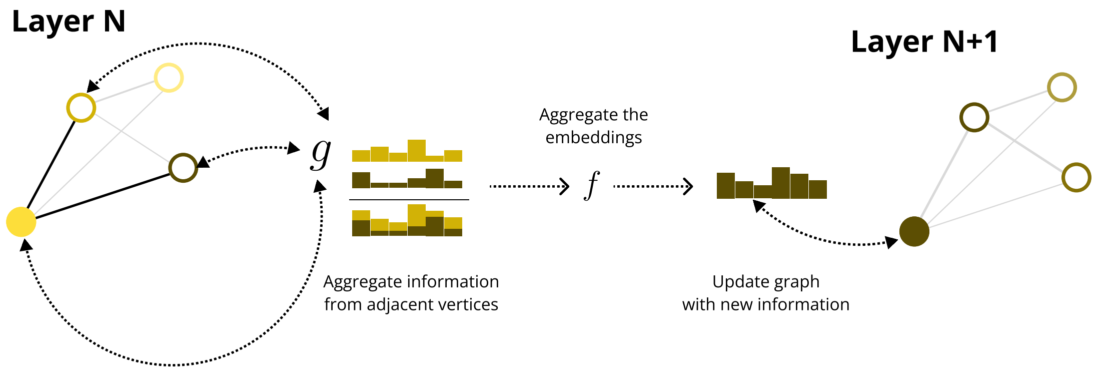

Layer N
g
Aggregate the embeddings
f
Layer N+1
Aggregate information from adjacent vertices
Update graph with new information

Figure 5: Message passing from the adjacent vertices of the filled vertex and the update of the embeddings in the next layer. For illustration purposes, the gathering function g is a simple stacking operation, and f is a summation.

operations that aggregate information from the neighbours of a specific element and use this information to update the element's value. By stacking message passing GNN layers, a vertex can incorporate information from multiple hops of its neighbouring vertices: i.e. a two-layered message passing GNN permits a particular vertex $v_i \in V$ to gather embeddings of $adj(v_i)$ and $adj(v_j)\ \forall v_j \in adj(v_i)$. We refer the reader to [41] for a detailed formalisation of the message passing framework. Some works that rely on message passing to generate embeddings are DimeNet [61], HC-GNN [161], LaMP [67], and PCAPass [110].

Graph Convolutional Networks (GCNs): extends the concept of convolution operations from images[12] to more complex topologies such as graphs. Recall that A and D are, respectively, the adjacency and the degree matrices of a certain graph G. We can define G's non normalised Laplacian as $L = D - A$. In this way, polynomials of the following form can be expressed:

$$p_w(L) = w_0 I_{|V|\times|V|} + w_1 L + w_2 L^2 + \cdots + w_d L^d = \sum_{i=0}^{d} w_i L^i$$

which can be thought as filters of a vanilla CNN with weights $w = [w_0, \ldots, w_d]$, where $I_{|V|\times|V|}$ is the identity matrix of dimensions $|V|\times|V|$. Having the vertex attribute matrix F, the convolution of F w.r.t. the polynomial above can be defined as $F' = p_w(L) \cdot F$. Therefore, convolving F with $p_w(L)$ of degree d means that, for each vertex v_i, its embedding gets influenced by at most d hops away. Thus, these polynomial filters are localised with d as the degree of the localisation. Defferrard et al. [31] extend the idea - hereafter ChebNet - of the Laplacian polynomial filters to consider degree-i Chebyshev polynomials of the first kind. Now, the d-degree Laplacian polynomial becomes $p_w(L) = \sum_{i=1}^{d} w_i T_i(\tilde{L})$ where $\tilde{L}$ is the normalised Laplacian according to the largest eigenvalue of L. By stacking ChebNet of layers with non-linear activation functions, one can perform a convolution over the input graph. In particular, if there are K different polynomial filters - denoting the network layers - the k-th layer has its own learning weights $w^{(k)}$. Hence, starting with $h_v^{(0)} = F[v]$ one can iterate for $k = 1, \ldots, K$ and compute $h_v^{(k)} = \sigma(p_{w^{(k)}}(L) \cdot h_v^{(k-1)})$ where $\sigma(\cdot)$ is a non linear activation function, and $h_v^{(k)}$ is the hidden representation of vertex v in layer $k = 1, \ldots, K$. Note that ChebNets reuse the same filter weights across different vertices, mimicking weight-sharing in CNNs. Let us focus on the convolutional operation of a specific vertex v_i via the polynomial kernel $p_w(L) = L$:

$$\begin{aligned}(L \cdot F)[v_i] &= L[v_i] \cdot F[v_i] \\ &= \sum_{v_j \in V} L[v_i, v_j] \times F[v_i] \\ &= \sum_{v_j \in V} (D[v_i, v_j] - A[v_i, v_j]) \times F[v_i] \\ &= D[v_i] \times F[v_i] - \sum_{v_j \in adj(v_i)} F[v_j]\end{aligned}$$

[12]Images can be seen as graphs with a regular grid-like structure, where the individual pixels are vertices, and the RGB channel values at each pixel as the vertex attributes.

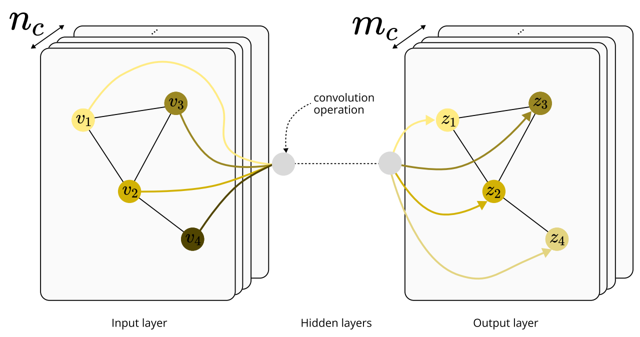


Figure 6: Illustration of a multi-layer GCN with n_c input channels and m_c feature maps in the output layer. The structure of the input graph (shown as the dotted lines) is shared in the hidden layers.

This is a 1-hop localised convolution. But more importantly, we can think of this convolution as arising of two steps: (1) aggregating over immediate neighbour attributes $F[v_j]\ \forall v_j \in adj(v_i)$, and (2) combining v_i's attributes $F[v_i]$. These convolutions can be thought of as "message-passing" between adjacent nodes: after each step, every node receives some "information' from its neighbours. By iteratively repeating the 1-hop localised convolutions K times - i.e. repeatedly "passing messages" - the receptive field of the convolution effectively includes all nodes up to K hops away. Figure 6 illustrates the general idea behind a GCN. Notice that the number of channels/filters can be different in the layers of the network, similarly as in a CNN. A GCN's layer-wise forward propagation works as follows:

$$h_{v_i}^{(0)} = F_{v_i}\ \forall v_i \in V$$
$$h_{v_i}^{(k)} = f^{(k)}\left(W^{(k)} \cdot \frac{\sum_{v_j \in adj(v_i)} h_{v_j}^{(k-1)}}{|adj(v_i)|} + B^{(k)} \cdot h_{v_i}^{(k-1)}\right)\ \forall k \in [1, K]$$

In other words, the embedding of vertex v_i at layer k is a linear combination of the mean of the embeddings of v_i's neighbours and v_i's own embedding at the previous layer, $k-1$. Predictions can be made at each vertex v_i by using the final compute embedding, $h_{v_i}^{(K)}$. We refer the reader to [60, 158] for a detailed explanation of GCNs. Some works that rely on graph convolutions to generate embeddings are MoNet [87], Feastnet [132], DeepInf [106], and PinSage [149].

2.1.3 Model interpretability

With the proliferation of deep learning and its use across various applications in society, trust has become a central issue. Given the black-box nature of these deep learning systems, there is a strong desire to understand the reasons behind their decisions. This has led to the sub-field of explainable AI (XAI) to gain prominence [47]. According to the works in [73, 107], explaining a model can (i) *give insight into the model's training and generalisation* or (ii) *give insight into the model's predictions*. The first category encompasses models that are inherently transparent - e.g. decision trees - and can provide indications of the path the model took to make its decision. However, the majority of interpretability strategies in deep learning fall into the second category: i.e. post-hoc interpretability. As suggested, post-hoc interpretability methods refer to the explanation of the outcome after the model has finished its training. Generally, models that are not intrinsically interpretable are denoted as black-boxes. Post-hoc interpretability engages in opening the black-box and explaining to domain laymen the reasons behind the predictions. Additionally, they can be exploited to reveal hidden patterns in the overall model behaviour. This type of interpretability approaches can be broken into more specific categories: i.e. counterfactual and feature-based explanations. Before delving into counterfactual explainability in graphs - discussed in Section 2.2 - we provide the reader with an overall description of a *counterfactual*. A counterfactual explanation for a prediction highlights the smallest change to the feature values that changes the prediction to a predefined output [86]. Here, we rely on the counterfactual formalisation provided in [44] (see Definition 1).

Definition 1. *Given a classifier Φ that outputs the decision $\Phi(x)$ for an instance x, a counterfactual explanation consists of an instance x' such that the decision $\Phi(x') \neq \Phi(x)$ and such that the difference between x and x' is minimal.*

In this context, we assume that the classifier Φ - also called *Oracle* - is a black-box model and the instances x and x' consist of a set of features $\mathcal{F}_x$ and $\mathcal{F}_{x'}$, respectively. Thus, counterfactual explanations belong to the family of example-based explanations [1]. Differently from counterfactuals, other example-based explanations [86] do not provide any insight on the way an instance x needs to mutate itself in order to change its prediction $\Phi(x)$. For example, let us consider that the customer x requests a bank loan which the intelligent system Φ rejects. A counterfactual explanation can reveal that a hypothetical customer x' would have the loan accepted, where x' is identical to x but with a yearly income of $\$50,000$ instead of $\$45,000$ and without any debt left. The hypothetical customer x' is a counterfactual example and the counterfactual explanation consists of the income of $\$50,000$. Thus, according to [44], a counterfactual explanation C is a set of counterfactual examples: i.e. $C = \{x'_1, \dots, x'_h\}$. Moreover, a counterfactual explainer is defined as follows:

Definition 2. *A counterfactual explainer is a function f_k that takes in input a classifier Φ, a set X of known instances, and a given instance of interest x. Hence, $f_k(x, \Phi, X)$ returns a set $C = \{x'_1, \dots, x'_h\}$ (s.t. $h \leq k$) of valid counterfactual examples where k is the number of counterfactuals required.*

Factual vs counterfactual explanations: Differently from counterfactual explanations, there are other techniques that base their explainability on the answer to the following question: *what were the dominant features that contributed to the outcome $\Phi(x)$ produced on the examined instance x?* These techniques are usually dubbed as *factual* or *attribute-based* explanations [154]. In this scenario, the explanation quantifies the impact of each feature on the outcome. Taking into consideration the previous example, the factual explanation for the loan being rejected could be that the yearly income is low and the debt left prior to the loan is high. Hence, income and debt are the features that mostly contributed to the rejection of the requested loan. Like counterfactual, most factual explainers also rely on comparing the examined instance x to one or more counterfactual inputs - referred as baselines. However, the role of counterfactuals here is to break apart the relative importance of features rather than to identify new instances with favourable predictions. For instance, SHAP [76] operates by considering counterfactuals that eliminate features and notes the marginal effect on the prediction. In other words, we note the change in the outcome when a feature is eliminated. We repeat the process for all combinations of features while computing a weighted average of the marginal effect on $\Phi(x)$.

2.2 Defining counterfactual explanations in graphs

In this Section, we discuss the graph counterfactual explanation definitions used in the literature. Notice that not all the works have provided a formal graph explanation. Most of them present a loss function, typically based on the distance between the prediction, $\Phi(v)$ and the original vertex v. Thus, for completeness purposes, first we discuss the original definition of the loss functions present in the literature. Then, we uniform the definitions according to the concepts and notation introduced in Section 2.1, and we discuss the advantages and drawbacks of each approach. Finally, we provide a unique uniform definition which encompasses all the others while discussing its motivation. Moreover, note that all the works presented in this survey treat the GCE problem as a binary classification task while the unified definition encompasses the general multi-class task.

2.2.1 Original counterfactual definitions in the literature

As mentioned above, the literature in GCE has tried to provide a formalisation for the problem definition and, eventually, the way a counterfactual example is defined. Because the definitions do not follow a rigorous formalisation, the works in this survey concentrate in providing an optimisation of a distance-based[13] loss function. Contrarily, only one work [11] does not encapsulate their definition into the optimisation of a certain loss function. The following works are presented according to their publication year, and we use the original notation used in them.

The authors of *"CF-GNNExplainer: Counterfactual Explanations for Graph Neural Networks"* [75] generate counterfactual examples by minimising the following loss function:

$$\mathcal{L} := \mathcal{L}_{pred}(v, \bar{v} | f, g) + \beta \mathcal{L}_{dist}(v, \bar{v}) \tag{3}$$

where v is the original vertex, f is the original model, g is the counterfactual model that generates $\bar{v}$ and $\mathcal{L}_{pred}$ is a prediction loss that encourages $f(v) \neq f(\bar{v})$. $\mathcal{L}_{dist}$ is a distance loss that encourages $\bar{v}$ to be close to v, and β controls how important is $\mathcal{L}_{dist}$ compared to $\mathcal{L}_{pred}$. The goal is to find $\bar{v}^*$ that maximises Equation 3, thus generating the optimal counterfactual example for v.

The authors of *"Model Agnostic Generation of Counterfactual Explanations for Molecules"* [141] entail that a counterfactual x' is specific to the example of interest x, where the authors have made a prediction $\hat{f}(x)$. The counterfactual is the explanation of x and defined by the solution to the following constraint optimisation problem:

$$\text{minimise } d(x, x') \text{ s.t } \hat{f}(x) \neq \hat{f}(x') \tag{4}$$

where x is the feature vector of the prediction, $d(x, x')$ is a measure of distance between features, and $\hat{f}(x)$ is the model.

The authors of *"Robust Counterfactual Explanations on Graph Neural Networks"* [11] are the only ones that provide a formal definition for GCE as follows.

[13]The function to be optimised can also be a similarity metric. In this case, the loss needs to be maximised instead.

Definition 3. *Given a GNN model Φ trained on a set of graphs D, for an input graph $G = \{V, E\}$, our goal is to explain why G is predicted by the GNN model as $\Phi(G)$ by identifying a small subset of edges $S \subseteq E$, such that (1) removing the set of edges in S from G changes the prediction on the remainder $\{V, E - S\}$ of G significantly; and (2) S is stable with respect to slight changes on the edges of G and the feature representations of the nodes of G.*

The authors of *"Counterfactual Graphs for Explainable Classification of Brain Networks"* [2] use the notion of counterfactual graph search to minimise the optimisation function in Equation 5. Given a black-box graph classifier $f : \mathbb{G}(V) \to \{0,1\}$ and a graph $E \in \mathbb{G}(V)$, find a counterfactual graph $E^* \in \mathbb{G}(V)$ such that:

$$E^* := \arg \min_{E' \in \mathbb{G}(V): f(E)=1-f(E')} d(E, E') \tag{5}$$

The authors of *"MEG: Generating Molecular Counterfactual Explanations for Deep Graph Networks"* [93] optimise a particular distance-based loss function in molecules that incorporates their structural information. Hence, given a target molecule m with prediction $\varphi(m)$, and given the counterfactual m', the authors rewrite their optimisation problem as follows:

$$\arg\max_{m'} \mathcal{L}(\varphi(m), \varphi(m')) + \mathcal{K}[m, m'] \tag{6}$$

where $\mathcal{L}$ is a measure of dissimilarity between the model predictions for the molecule m and its counterfactual m', while $\mathcal{K}$ measures the structural similarity between molecules m and m'.

For completeness purposes, we also report the factual-based formulation of *"Learning and Evaluating Graph Neural Network Explanations based on Counterfactual and Factual Reasoning"* [124] because the authors claim to perform factual and counterfactual explanations simultaneously. Note that the scope of the original paper is to provide a factual explanation which can be used to derive a counterfactual example from it. The proposed method optimises factual search (see Eq. 7) by considering counterfactual properties (see Eq. 8) thus since the method do not generate directly counterfactual we cannot provide a unified formalisation as done in Section 2.2.2.

$$\arg\max_{c \in C} P_\Phi(c \mid A \cdot M, X \cdot F) = \hat{y} \tag{7}$$

$$\arg\max_{c \in C} P_\Phi(c \mid A - A \cdot M, X - X \cdot F) \neq \hat{y} \tag{8}$$

where A is the adjacency matrix, X is the vertex feature, M is the learnt edge mask (i.e. $M \in \{0,1\}^{V \times V}$, and F is a learnt attribute mask (i.e. $F \in \{0,1\}^{V \times d}$ s.t. d is the number of vertex attributes). According to the equations introduced above, factual reasoning seeks a sub-graph whose information is sufficient to produce the same prediction as using the original graph. Contrarily, counterfactual reasoning seeks a sub-graph whose information is necessary which if removed will result in different predictions. Notice that, although the method proposed in [124] generates factual explanations, according to the employed optimisation function, the counterfactual example can be considered as the remainder of the original graph when the factual sub-graph[14] gets eliminated. Moreover, by concentrating first in identifying the subset of edges, vertices, and vertex attributes to form a sub-graph for the factual explanation, the authors do not guarantee that the counterfactual sub-graph is minimal.

2.2.2 Discussion and unification of the GCE definitions in the literature

Before providing a formal definition of graph counterfactual explanations and its characteristics, we first unify, for consistency and readability purposes, the notation of Section 2.2.1. Furthermore, we discuss the drawbacks of each of them and provide the reader with insights that we exploit in our proposed definition (ref. Section 2.2.3). Notice that, throughout this Section, we use $\mathcal{S}_{pred}(\Phi(x), \Phi(x'))$ to indicate a similarity function between the outcome of the original instance $\Phi(x)$ and the counterfactual one $\Phi(x')$. Additionally, we denote with $\mathcal{S}_{inst}(x, x')$ a similarity function between the original instance x and the counterfactual one x'. For completeness purpose, we define $\mathcal{D}_{pred}(\Phi(x), \Phi(x'))$ and $\mathcal{D}_{inst}(x, x')$ as distance functions representing the counterparts of $\mathcal{S}_{pred}$ and $\mathcal{S}_{inst}$, respectively.

As described in Equation 3, Lucic et al. [75] exploit the predictive model f and the counterfactual generative model g to minimise the distances between the outcomes, $f(v)$ and $f(g(v))$ (i.e. $\mathcal{L}_{pred}(v, \bar{v}|f, g)$), respectively, and the distance between the original instance v and the generated counterfactual $\bar{v} = g(v)$ (i.e. $\mathcal{L}_{dist}(v, \bar{v})$). Here, the authors generate counterfactuals over the vertices $v \in V$ for the graph in input. According to the original paper, $\mathcal{L}_{pred}$ is defined as the multiplication of the negative log-likelihood (NLL) and a negated indicator function with condition $f(v) = f(\bar{v})$. Thus, the chosen formulation for $\mathcal{L}_{pred}$ admits a non-counterfactual example as a possible generated example, including the original one that minimises the overall loss $\mathcal{L}$. In other words, if $\bar{v} = v$, then both $\mathcal{L}_{pred} = 0$ and $\mathcal{L}_{dist} = 0$. In this

[14]Notice that the factual sub-graph contains also vertex attributes.

case, the original v is also a minimal "counterfactual" example. Finally, the loss function provided in [75] becomes the following:

$$\arg\min_{x' \in X'} \mathbb{1}[\Phi(x) = \Phi(x')] \cdot \mathcal{D}_{pred}(\Phi(x), \Phi(x')) + \beta \cdot \mathcal{D}_{inst}(x, x') \tag{9}$$

where $\mathbb{1}[\Phi(x) = \Phi(x')]$ is an indicator function producing 1 if $\Phi(x) = \Phi(x')$ and 0 otherwise.

Wellatte et al. [141] minimise the distance between the original instance x and the counterfactual one, x', under the condition that their outcomes are different (i.e. $f(x) \neq f(x')$) - see Equation 4. The minimisation constraint makes them robust towards the pitfall of the previous formalisation. In other words, because the outcomes $f(x)$ and $f(x')$ need to be different, the instance x' in the calculation of the distance needs to be from another class w.r.t. $f(x)$, thus $x' \neq x$. Because the authors use similarity and distance function interchangeably[15], one should seek to maximise, rather than minimise, the similarity between x and x'. Hence, the optimisation function provided in [141] becomes the following:

$$\arg\max_{x' \in X' \mid \Phi(x) \neq \Phi(x')} \mathcal{S}_{inst}(x, x') \tag{10}$$

Although being the only work that provides a formal definition of a robust counterfactual explanation, Bajaj et al. [11] - see Definition 3 - cannot always change the prediction of the model by just removing edges. In general, it is not always possible to find a set of edges $S \subseteq E$ that maintains the stability w.r.t. changes on G. Thus, in these cases, generating a counterfactual example is impossible. According to the definition the authors provide, the set $E - S$ contains the factual edges for which $\Phi(G = (V, E - S)) \neq \Phi(G = (V, E))$. Without loss of generality, all definitions and/or implementation methods that generate counterfactual examples by only removing edges suffer from the same phenomenon.

As provided in Equation 5, Abrate and Bonchi [2] minimise the distance function $d(E, E')$ of the two graphs E and E' (counterfactual). Having fixed the set of vertices, the authors generate a counterfactual graph E' that shares the vertices of E and that minimises the symmetric difference between the edges of E and E'. Their optimisation function takes into consideration a similar constraint to that introduced in Equation 10. However, the main difference here is that the outcome of the counterfactual has to be exactly specular to that of the original instance, confining it to a mere binary classification task. Therefore, we write the optimisation function provided in [2] as follows:

$$\arg\min_{x' \in X' \mid \Phi(x) = 1 - \Phi(x')} \mathcal{D}_{inst}(x, x') \tag{11}$$

Finally, Numeroso and Bacciu [93] exploit both a distance between the outcomes of the original molecule m and the counterfactual one, m', and a similarity between the structure of the two molecules, $\mathcal{K}[m, m']$ (see Equation 6). According to the original paper, when performing a classification, a model φ emits a probability distribution $\varphi(\cdot) = y = [y_0, \ldots, y_{|C|}]$ for a certain set of classes C. Hence given a molecule in input m and its outcome $\arg\max_{c \in C} \varphi(m)$, the authors produce counterfactual molecules m' that maximise the outcome of classes different from c, and the similarity between their structures. The final form of the proposed optimisation function maximises the negation of the probability of outputting a class equal to c (i.e. y_c) and $\mathcal{K}[m, m']$ controlled by the weight α and $1 - \alpha$, respectively. Maximising the negation of the probability of producing c is fancy wording for maximising the distance between the outcomes on the original instance x and the counterfactual x'. It is important to highlight that the counterfactual can be generated by adding and removing both vertices and edges. Hence, the counterfactual graph is not necessarily a sub-graph of the original one. Finally, the formalisation in [93] becomes the following:

$$\arg\max_{x' \in X'} \alpha \cdot \mathcal{D}_{pred}(\Phi(x), \Phi(x')) + (1 - \alpha) \cdot \mathcal{S}_{inst}(x, x') \tag{12}$$

Notice that this formalisation is a generalisation of Equation 10 because it introduces $\mathcal{D}_{pred}(\Phi(x), \Phi(x'))$ instead of the constraint $\Phi(x) \neq \Phi(x')$. Additionally, the weighing factor α decides which measure contributes more to the generated optimal counterfactual x'.

2.2.3 Minimal graph counterfactual explanations

Considering the variety of the proposed GCE definitions presented above, we find it necessary to define a comprehensive one which, on the one hand, encompasses all of them and, on the other, ensures their correctness and provides enough flexibility to embed future definitions. To do so, in this Section, we introduce a formal definition and provide comments on its characteristics and main differences from the approaches presented in Section 2.2.2. Notice that, regardless of the generation operation adopted to produce the counterfactual examples, we obtain a resulting graph explanation G'. The

[15]In their original implementation of the optimisation function, Wellatte et al. [141] exploit the Tanimoto index as a similarity measure between x and x'.

generation operation might encompass adding/removing vertices and edges from the original graph as well as tweaking their attributes.

First, we present a more general setting of our GCE definition taking into consideration the graph classification problem (see Definition 4). Then, we specialise it to take into consideration vertex and edge classification by slightly modifying the prediction model Φ. Notice that we use the notation x to describe an instance (i.e. vertex or edge) of the original graph G, and $G' \in \mathcal{G}'$ to describe the counterfactual examples where $\mathcal{G}'$ is a set of all possible counterfactual examples.

Definition 4. *Multi-class minimal counterfactual examples: Let Φ be a prediction model that classifies G into a class $c \in C$ from a set of classes C. Let $\mathcal{G}'$ be the set of all possible counterfactual examples G' and $\mathcal{S}_{inst}(G, G')$ be a similarity measure that tells how similar G' is to G. Then, we define the set of counterfactual examples w.r.t. Φ as follows:*

$$\begin{aligned} s(c', G) &:= \max_{G' \in \mathcal{G}', G \neq G'} \{\mathcal{S}_{inst}(G, G') \mid \Phi(G') = c'\} \\ \mathcal{E}_{\Phi}(G) &:= \bigcup_{c' \in C - \{c\}} \{G' \in \mathcal{G}' \mid G \neq G', \mathcal{S}_{inst}(G, G') = s(c', G)\} \end{aligned} \tag{13}$$

According to Equation 13, $\mathcal{E}_{\Phi}(G)$ has the maximally similar counterfactual examples $G' \in \mathcal{G}'$ to the original graph G for each class c' that is different to the prediction on G (i.e. $c' \in C - \{c\}$ where $c = \Phi(G)$). More specifically, we find those counterfactual examples that must be different from the original graph G such that they maximise a similarity function w.r.t. the original graph $\mathcal{S}_{inst}(G, G')$. Accordingly, we can refer to the counterfactual examples of a specific class $c' \in C - \{c\}$ as $\mathcal{E}_{c',\Phi}(G) = \{G' \in \mathcal{G}' \mid G \neq G', \mathcal{S}_{inst}(G, G') = s(c', G)\}$.

The definition above has two main advantages w.r.t. the ones provided in the literature: i.e. (1) it supports all the graph-based tasks in a multi-class scenario, and (2) it contains all the minimal counterfactual examples for each class $c \in C$. Furthermore, differently from the majority of the formalisations in Section 2.2.2, without loss of generality, we found that using a similarity function $\mathcal{S}_{inst}(G, G')$ is more beneficial because of the flexibility to consider both the attributes and the structure of the graph (i.e vertex and edge attributes, and edge connections). Notice that, it is always possible to define the similarity measure as $\mathcal{S}_{inst}(G, G') = 1 - \widetilde{\mathcal{D}}_{inst}(G, G')$ where $\widetilde{\mathcal{D}}_{inst}(G, G')$ is a normalised distance function in [0,1]. Finally, we constraint the counterfactual example to be different to the original graph (i.e. $G' \neq G$) because we decouple the definition from the performances[16] of the model Φ.

Definition 5. *Global minimal counterfactual example: Let Φ be a prediction model that classifies G into a class $c \in C$. Let $\mathcal{G}'$ be the set of all possible counterfactual examples G'. To find the global minimal counterfactual example $\mathcal{E}^*_{\Phi}(G)$ of the original graph G, we define it as follows:*

$$\mathcal{E}^*_{\Phi}(G) = \arg \max_{G' \in \mathcal{G}', \mathcal{S}_{inst}(G, G')} \mathcal{G}' \tag{14}$$

As anticipated at the beginning of this Section, we can extend Definition 4 to take into consideration also vertex and edge classification tasks. The only component that changes is the prediction model Φ. Recall that we denote with x an instance - being either a vertex or an edge - belonging to the original graph G. In this way, the prediction model Φ takes in input the instance x and G to produce a class c (i.e. $\Phi(x, G) = c$). Hence, for a particular counterfactual example graph G', $\Phi(x, G') = c'$. It is important that the generated counterfactual $G' = (V', E')$ contains the original instance x: i.e. in case of vertex classification, $x \in V \ \wedge \ x' \in V'$; whereas, for edge classification, $x \in E \ \wedge \ x' \in E'$. In this way, we can understand how x's relations (vertices or edges) in its vicinity have changed in G' w.r.t. G.

3 Methods

Here, we present the Graph Counterfactual Explanations methods in the literature by summarising them into a classification of eight dimensions. In Section 3.1, we show, through a complete taxonomy, where the counterfactual explainability methods reside w.r.t. the other graph explanations techniques, discussing their main properties. In Section 3.2, we present the eight chosen dimensions and summarise all the works in a concise table. Finally, in Section 3.3, we discuss every method by briefly describing them and examining how they belong to each of the previously identified eight dimensions.

3.1 Where does counterfactual explainability stand in graph explanations?

The literature proposes many criteria for classifying explanation methods. One of the proposed categorisations distinguishes the methods between those that work at the instance-level (i.e. local) and those that work at global-level

[16]Imagine having Φ as a random classifier in $C = \{0, 1\}$ with probability 0.5. In this scenario, $\Phi(G')$ can be equal to $\Phi(G)$ if $G = G'$, which defies the rule of a counterfactual example G'.

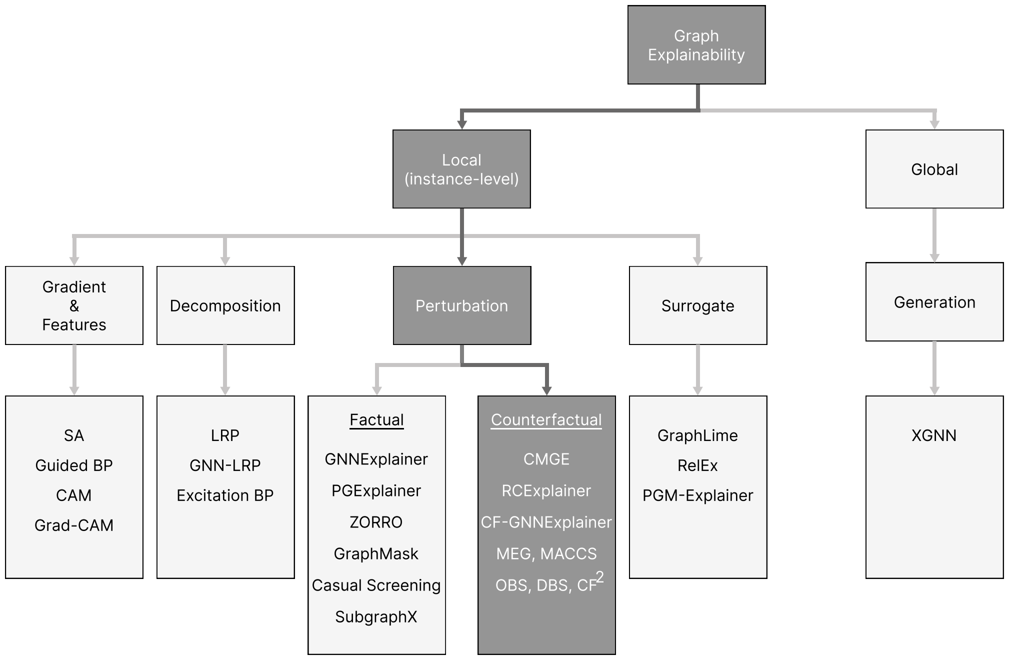


Figure 7: Where is graph counterfactual explainability placed according to the categorisation under the general graph explainability? The highlighted portion of the taxonomy describes the path towards counterfactual explainability treated in this survey. Figure adapted from the taxonomy provided in [153].

explainability [44, 153, 6, 9, 133]. The former focuses on providing the reasons that make a black-box model reach a specific decision on a particular input instance. Whereas, the latter entails producing an explanation of the overall logic of the black-box model assuming that the provided explanation is complete and valid for any instance. In other words, global-level explainability on graphs engages in providing the boundaries of the vector space of explained outcomes. Therefore, having a group of graph instances, global-level explainers aim to determine their collocation in the decision vector space. Only a handful of works focus on graph global-level explanations [152]. Differently, all methods in GCE belong to the class of local explainability.

To better understand GCE methods, it is important to highlight their relationship to other explanation techniques in graph data. In Figure 7, we provide the reader with a taxonomy of graph explainability approaches (of any kind) proposed in the literature. Notice that, under the local explainability class, there are four different techniques to generate explainable examples. In this survey, we only treat perturbation-based methods because counterfactual explainability resides within this particular class. We refer the reader to [153] for a complete overview of the other methodologies.

Without loss of generality, perturbation-based methods study the output variations w.r.t. input changes. Both factual and counterfactual perturbation-based methods follow a similar high-level pipeline consisting of 1) generating masks that indicate features of interest given a specific input graph G; 2) combining the mask with G to obtain a new graph G' such that the features of interest remain unchanged; 3) feeding G' to the prediction model Φ and updating the mask according to the outcome $\Phi(G')$. In factual explainability, the goal is to obtain a mask that induces a sub-graph G' of G such that $\Phi(G') = \Phi(G)$. Intuitively, G' only contains the most important features according to the prediction task at hand. Contrarily, counterfactual explainability searches for a graph G' such that $\Phi(G') \neq \Phi(G)$ and the $\mathcal{D}_{inst}(G, G')$ is minimal (see Definition 5). In other words, G' contains a minimal number of perturbations w.r.t. the original graph G.

3.2 GCE literature classification

Here, we follow the reasoning behind the classification of the literature as done similarly in [46, 44, 8]. Notice that we provide the reader with a concise classification of the GCE methods proposed in the literature by identifying more dimensions w.r.t. the other surveys, as some of these are tightly related to the graph domain. Hence, Table 3 illustrates the eight dimensions (i.e. model agnosticism, model accessibility, factual-based, minimal counterfactual example (CE), domain agnosticism, training data accessibility, classification task, and generation type) according to which we categorise each of the works analysed in this survey. For each of them, we provide a brief description and, then, delve into further detail. Recall from Section 2.1.3 that there are two categories of interpretability/explainability strategies.

Table 3: Comparison of the main characteristics of GCE methods. $\times$ depicts a missing aspect; ✓depicts a covered aspect; $\sim$ depicts a partially covered aspect.

Method	Model Agnosticism	Model Access	Factual-Based Explanations	Minimal CE	Domain Agnosticism	Training Data Accessibility	Classification Task	Generation Type
CMGE [145]	$\times$	✓	✓	$\times$	$\sim$	✓	Graph	Perturb: $E(+,-), V(-)$
RCExplainer [11]	✓	✓	✓	$\times$	✓	✓	Graph, Vertex	Perturb: $E(-)$
CF-GNNExplainer [75]	✓	✓	$\times$	✓	✓	$\times$	Vertex	Perturb: $E(-)$
MEG [93]	✓	✓	$\times$	✓	$\times$	$\times$	Graph	Perturb: $E(+,-), V(+,-)$
MACCS [141]	✓	$\times$	$\times$	✓	$\times$	$\times$	Graph	Perturb: $E(+,-), V(+,-)$
BOS [2]	✓	$\times$	$\times$	✓	✓	$\times$	Graph	Perturb: $E(+,-)$
BDS [2]	✓	$\times$	$\times$	✓	✓	✓	Graph	Perturb: $E(+,-)$
CF^2 [124]	✓	$\times$	✓	$\times$	✓	$\times$	Graph, Vertex	Perturb: $E(-), V(-), F(-)$

Notice that all methods for GCE belong to the post-hoc explainability category. Hence, the literature that we revise in this survey encompasses only black-box models. In Section 3.3, we describe the surveyed methods and discuss their approaches considering the vision reported in Table 3. Where we organised - to give the reader an index of finding the method that best suits their needs - the explainability methods according to minimal counterfactual examples.

Below we report a detailed description of the eight dimensions that we identified.

Model Agnosticism - As introduced in Section 2.1, a graph counterfactual explainer gets a black-box model Φ in input and generates counterfactual examples by maximising either Equation 13 or 14. Notice that the desiderata for a counterfactual explainer is to decouple its procedure of producing counterfactuals G' over an input instance G from the prediction $\Phi(G')$ s.t. $\Phi(G) \neq \Phi(G')$. In other words, model-agnostic counterfactual explainers can be exploited to explain the outcome of any prediction model Φ. Contrarily, model-specific explainers are intrinsically reliant on a particular class of prediction models (e.g. attention-based methods).

Model Access - GCE methods might require different levels of access to the underlying ML model whose decision they are explaining. Inspired by Verma et al. [133], we can define two levels of model access: i.e. embedding-wise and gradient-wise access. Embedding-wise access entails that the explainer method has the possibility to grab the embedding of the input graph G at a specific layer of the underlying model Φ or use an arbitrary method to generate the embeddings. Whereas, gradient-wise access consists of obtaining the gradients of G at any specific layer of the model Φ. Without loss of generality, contrarily to gradient-wise access, embedding-wise access does not restrict the explainer to be reliant only on neural network architectures.

Factual-based explanations - Counterfactual explainers can take into consideration a feature/factual-based explanation to generate the counterfactual examples. In other words, this class of explainers uses a factual explainer to generate a factual example of the input graph G producing the most important features w.r.t. the prediction $\Phi(G)$. Then it is possible to build a counterfactual example G' by changing the most important features that entail a change in the overall outcome $\Phi(G')$. Factual-based counterfactual explainers do not guarantee to produce a minimal counterfactual example because they do not use an optimisation function to minimise the distance between the generated example G' and the input G. Figure 8 illustrates three different explanations. On the left corner, we depict a factual example - highlighted connections - representing the most important characteristics of the Nitrobenzene molecule. The middle "*“molecular structure”* illustrates a derived counterfactual example from the factual one on the left. It is not guaranteed that the derived counterfactual example entails a valid molecule. Without loss of generality, derived counterfactuals can only be produced if the original graph was perturbed by eliminating vertices/edges (see Section 3 for more details). For completeness purposes, we illustrate tse non-minimality of factual-based counterfactual explainers (compare the molecule on the middle with the one on the left). Here, we can clearly see that the minimal counterfactual has the three connections highlighted that, if removed, would break the Nitrobenzene structure shown on the left, thus, avoiding it to be a carcinogen.

Minimal Counterfactual Example (CE) - According to the definitions provided in the literature of counterfactual explainability, generating examples G' which have minimal changes w.r.t. the input G is a crucial aspect. Additionally, Guidotti [44] enlists minimality as one of the first criteria that a counterfactual example should have. Providing a minimal counterfactual example is useful when giving the end-user tangible explanations. For instance, recall the loan approval example. Let us assume that the counterfactual example was *if the customer were a millionaire, then her loan would be granted*. Notice that this particular example is a useless counterfactual since it does not provide a meaningful explanation. Minimal counterfactual examples ensure that the change in the outcome is guaranteed when the structure of the input slightly differs. In the loan approval example, a minimal counterfactual explanation would be *if the customer had $1,000 more in her bank account, then her load would be granted*, and the minimality is represented as the additional $1,000.

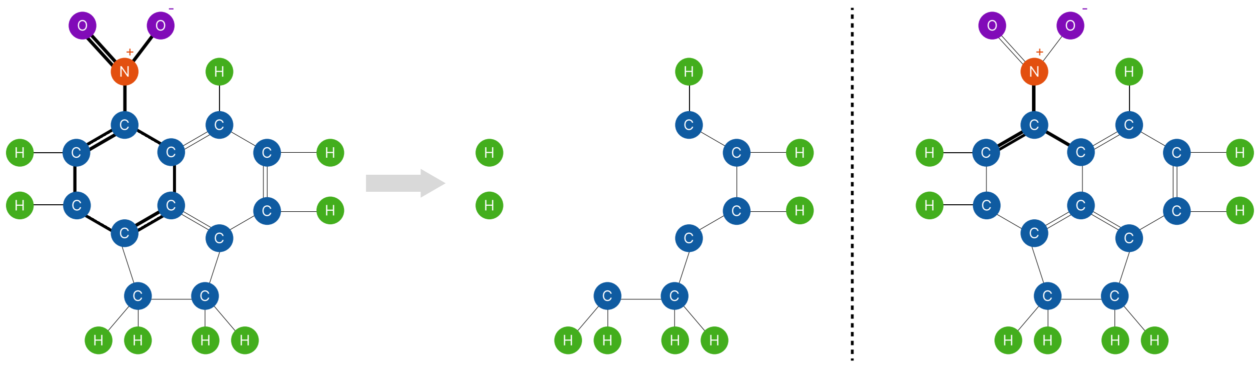


Figure 8: An example of a 5-Nitroacenaphthene molecular structure. (left) Factual example, represented as bold connections, as a Nitrobenzene molecule. (middle) Derived counterfactual example by eliminating the Nitrobenzene molecule. Notice that the induced counterfactual example might not make a valid molecule structure. (right) Minimal counterfactual example highlighting the three connections that, if removed, break the Nitrobenzene structure avoiding the molecule to be a carcinogen.

Domain Agnosticism - As the name suggests, domain-agnostic counterfactual explainers are transferable from one domain to the other. In particular, this kind of explainers can work with heterogeneous graph data (e.g. passing from molecular graphs to human social networks). On the other hand, domain-specific explainers are specialised in generating counterfactual examples on only a subclass of graph data (e.g. protein-protein interactions). These explainers, when given in input a graph from a different domain, fail to generalise or even output a valid counterfactual example. The desiderata for robust counterfactual explainers is to be domain-agnostic and transcend themselves from properties valid only in specific cases.

Training Data Accessibility - Counterfactual explainers can access training data to generate meaningful examples or be oblivious and rely only on the prediction of the model Φ while maximising the similarity between the counterfactual G' and the input graph G. When trying to preserve privacy in critical domains, accessing further data is not feasible. In these scenarios, we can estimate how the prediction model works by randomly perturbating G to produce a new graph G' and verifying whether the decision of the model changes (i.e. $\Phi(G') \neq \Phi(G)$). In this way, it is possible to construct an ad-hoc synthetic dataset of graph instances and understand the model's functioning while maintaining the original data's privacy. Hence, we can conclude that being oblivious of any additional information besides the input graph G is a desired feature for GCE methods.

Classification Task - In general, there are three classification tasks in graph learning: i.e. vertex, edge, and graph classification. In the literature of GCE, graph classification is the most common task. Additionally, edge classification has not been explored yet. However, the works studied in this survey can be adapted to perform this particular task.

Generation Type - The literature focuses on two main strategies of generating counterfactual examples: i.e. search-based and perturbation-based generation. Search-based methods find the most similar instance G' from the original dataset w.r.t. the input graph G s.t. $\Phi(G) \neq \Phi(G')$. This way of searching for a counterfactual example does not guarantee that the produced explanation is minimal since the searched examples have unchanged features. Contrarily, perturbation-based generation approaches perturb the input features of G and generate a new counterfactual example G'. In case perturbation-based methods minimise an optimisation function, they also guarantee to produce a minimal counterfactual explanation. Perturbations can happen on vertices, edges, and vertex/edge attributes. In cases of vertices and edges, the perturbation operations are either adding or removing them from the input graph. Meanwhile, since vertex/edge attributes can be real numbers, perturbing them can be done by adopting any kind of mathematical function. Finally, notice that a shortcoming of search-based approaches is that the explainer needs to access either the training data or the embeddings/gradients of the model Φ. In the Table we indicate with E, V, and F, the perturbation of the edge, vertex, and vertex attribute sets, respectively. Moreover, notice that + and - indicate, respectively, adding and removal operations over the set that precedes them in the notation.

3.3 GCE Methods

Here, we provide the reader with a detailed description of the approaches in GCE and a discussion on how they cope with the dimensions provided in Table 3. In particular, we first describe the method and its strategy of generating counterfactual examples. Then, we provide, for each dimension, the advantages and limitations of the method described. We invite the reader to notice that the first three methods do not provide any insight on the minimality aspect of counterfactual examples. Additionally, this Section does not explicitly treat the way the methods rely on the definition of counterfactual examples since was already extensively discussed it in Section 2.2.

Counterfactual Multi-Granularity Supporting Facts Extraction (CMGE) [145] is an explanation method designed to work on Electronic Medical Records (EMR). First, the authors transform the medical data into a hierarchical graph structure that encodes the relationship between the different types of records. Then, they train a learnable soft-mask matrix to mask the features of vertices/edges in the graph while keeping the decision unaltered. The remaining features - i.e. those that have not been masked - can be considered as supportive to the decision representing a particular health diagnosis. The authors rely on Graph Attention Networks (GATs) as the explainer of their approach despite the fact that they are not intended as interpretable models. In particular, Jain et al. [52] argues that attention mechanisms, in general, are wrongly being used in explainability while their original task is to enforce the prediction model to focus on some particular features that are regarded as important. When generating counterfactual examples, the authors focus on graph classification. The authors use these counterfactual examples as supportive facts when classifying the EMR instances.

1. *Model agnosticism and model access*: Because CMGE relies on the attention mechanism as an explanation method, the explainer and the oracle cannot be decoupled. Thus, CMGE becomes an oracle-specific explanation method. Additionally, CMGE requires complete access to the model logic because it needs to verify the attention weights for each feature to provide a possible explanation.

2. *Factual-based explanations and minimal CE*: Despite the authors calling their method counterfactual reasoning, it is actually a perturbation-based factual explanation method. The logic behind CMGE is to change the features of the input instance such that to obtain a counterfactual example. However, using this approach, the generated counterfactual examples are in general non-minimal. Here, by relying on the attention mechanism, the authors check which are the k most important features. By doing so, it is not guaranteed that this set of features is the smallest possible that generate counterfactuals. Additionally, it can happen that the set of k most important features misses others that belong to a counterfactual example w.r.t. a specific input G.

3. *Domain agnosticism and training data accessibility*: The original paper does not make any claims w.r.t. to domain agnosticism, but in the experimental phase, the authors concentrate only in assessing the performances of CMGE on EMR data. In principal, GATs can be exploited in other domains while maintaining the same structure as long as the data can be represented as a graph. Hence, Table 3 for CMGE has a $\sim$ symbol to indicate that it is possible to be domain-agnostic, but the authors fail to address this aspect. According to what we discussed in point 1, the explainer is oracle-specific since it ties itself to the attention mechanism provided. Hence, because the attention mechanism needs access to the weights of the network to produce contribution scores for each of the features, CMGE needs to have access to the training data to generate counterfactuals.

4. *Classification task and generation type*: The authors generate the counterfactual examples while performing a graph classification task. Additionally, they perform perturbations on the vertices and edges to engender the counterfactuals. Recall that CMGE is a factual-based counterfactual explainability method. Therefore, in the scenario of vertex perturbations, the authors can only remove vertices because they want to find a sub-graph whose outcome is the same as the graph in input. Then, by eliminating this subgraph from the original graph structure, the remainder could be a counterfactual example. For edge perturbations, the authors use both adding and removal operations to generate counterfactuals.

CF-GNNExplainer[75] finds a binary perturbation matrix that sparsifies the adjacency matrix of the input graph. To find the perturbation matrix the authors build on top of the method proposed in [119] for training sparse neural networks where the objective is to zero out the network weights. Contrarily, CF-GNNExplainer's objective is to zero out entries in the adjacency matrix (i.e. remove edges). As shown in Section 2.1, CF-GNNExplainer tries to minimise Equation 3, where the loss function makes use of the negative log-likelihood and is designed for the loss to be active as long as the original instance and the generated example belong to the same class. Then, the explainer generates the counterfactual example whose distance is the smallest w.r.t. the original graph.

1. *Model agnosticism and model access*: CF-GNNExplainer is a model-agnostic method that can be used with any oracle. To keep their model-agnosticism, the explainer does not access any information regarding the inner workings of the oracle, but instead, just needs the input instance and the oracle decision.

2. *Factual-based explanations and minimal CE*: The proposed approach is specifically designed for producing counterfactual examples and it is not based on existing factual explanation techniques. Moreover, CF-GNNExplainer is designed to tackle the minimal GCE problem, returning, from the generated counterfactual examples, the closest one to the input instance. However, due to the kind of perturbations performed, it is not always possible to find a minimal counterfactual.

3. *Domain agnosticism and training data accessibility*: This explanation method doesn't use any domain knowledge, so can be employed on different application domains. Furthermore, it does not require to be trained using labelled data.
4. *Classification task and generation type*: CF-GNNExplainer is focused on producing GCE for oracles trained for the vertex-classification task. For this reason, the explainer considers as input a given vertex and the subgraph of nodes and edges relevant for that node. While other methods focused on vertex counterfactuals perturb the vertex features, CF-GNNExplainer perturbs the structure around the node until the oracle decision changes. In other words, the authors want to change the relationships between instances, rather than change the instances themselves. Regarding the generation type, this approach only removes edges from the original graph, this is one of its main limitations as it is not always possible to obtain a GCE by just removing edges from the original instance.

Model Agnostic Counterfactual Compounds with STONED (MACCS) [141] works in the molecular domain. The method takes in input a molecular graph represented as SELFIES (*SELF*-referenc*I*ng *E*mbedded *S*trings) [64]. The approach employed for molecular counterfactual generation is built on the Superfast Traversal, Optimization, Novelty, Exploration and Discovery (STONED) method which enables rapid exploration of chemical space without a pre-trained generative model or set of reaction rules [90]. The STONED protocol consists of string insertion, deletion, and modification steps that can generate thousands of perturbations of a given molecule engendering valid molecules that are close in the chemical space. This method works because the molecules are represented as SELFIES whose modification/perturbation entails a valid molecule as well [64]. After expanding the chemical space around the original molecular graph, MACCS identifies similar counterfactual molecules with a changed prediction, selecting a small number of these using clustering and Tanimoto similarity. By clustering the counterfactual examples and selecting, for each cluster, the closest counterfactual to the original molecule, the explainer is capable of returning multiple counterfactuals that are different from each other.

1. *Model agnosticism and model access*: The explainer receives the model to be used as a parameter and does not make any particular assumption about it being completely model agnostic. Neither, the STONED method, or the clustering and similarity calculation steps, need to access the oracle internal information to produce and select the GCE.
2. *Factual-based explanations and minimal CE*: MACCS is not based on any factual explanation approach, being specifically designed to produce counterfactual examples. The explainer returns multiple counterfactual examples ordered by its closeness to the original molecular graph. Furthermore, the authors restricted the number of modifications in the counterfactual examples to a maximum of 2 by default. The aim of this restriction is to ensure the GCEs stay local in chemical space and being able to provide minimal counterfactuals.
3. *Domain agnosticism and training data accessibility*: By employing the STONED method, the explainer can generate GCEs without needing labelled training data. On the other hand the explainer is domain specific, being restricted to the molecular domain. The main limitation of MACCS is that the input graphs have to be in SELFIES representation, requiring them to be valid molecules. Also, the STONED method performs specific mutations, restricted to an alphabet specific to the molecular domain.
4. *Classification task and generation type*: MACCS is designed to explain the decisions of oracles focused on molecular graph classification. The reason of this selection is that usually in the molecular domain graphs are small and has more sense to classify a molecule given its properties than classifying an atom or a link between atoms. For the counterfactual generation process STONED can perform token deletions, replacements, and insertions of the SMILES, thus being able to modify vertices and edges of the original graph.

***RCExplainer* [11]** is a method to generate robust GCE. On a first step, the method models the decision logic of a GNN employing a set of decision regions, each induced by a set of linear decision boundaries of the GNN. RCExplainer uses an unsupervised method to find the decision regions for each class such that each decision region governs the decision on multiple graph samples predicted to belong to the same class. The linear boundaries of the decision region capture the common decision logic on all the graph instances inside that decision region, so they do not easily overfit the noise of an individual graph instance. In this way the produced GCEs are more robust to noise.

Then, on a second step, the method uses a loss function, based on the linear boundaries of the decision region, to train a neural network for producing robust GCEs as small subsets of edges of an input graph. The loss function is designed to optimise the explainability and counterfactual property of the subset of edges. In this way, the subgraph induced by the edges lies within the decision region, producing a prediction consistent with the input graph. Moreover, deleting the subset of edges from the input graph produces a subgraph that lies outside the decision region.

1. *Model Agnosticism and Model Access*: Regarding model access, RCExplainer extracts the decision region of a GNN in the d-dimensional output space of the last convolution layer of the GNN. Because the features

generated by the last convolution layer are more conceptually meaningful and more robust to noise than those raw features of input graphs, such as vertices and edge. The explainer assumes the oracle is a GCN, so it can access the last convolutional layer to obtain a vector representation of the graph. However, it can be adapted to work with other kinds of oracles as long as they provide a vector representation of the graph.

2. *Factual-based explanations and minimal CE*: RCExplainer aims to find a subset of edges of the input graph such that the prediction on the subgraph induced by these edges remains the same as in the input graph (factual explanation). However, also impose the condition that if the produced set of edges is removed from the input graph then the prediction should change (counterfactual explanation). Unifying the process of producing factual and counterfactual explanations has as a consequence that in general the method is not able to produce minimal GCEs.
3. *Domain agnosticism and training data accessibility*: The explanation method is not dependant of specific domain knowledge and can be used in diverse application domains. RCExplainer needs a collection of labelled instances to determine the decision regions governing each of the classes predicted by the oracle.
4. *Classification task and generation type*: The method is designed to explain the decision of oracles focused on the graph classification task. The GCE generation method of RCExplainer is based on perturbation over the edges of the input graph. However, it only considers edge removals, which can limit the ability of the method for finding counterfactual examples.

Oblivious Bidirectional Search (OBS) and Data-Driven Bidirectional Search (DBS) [2] are heuristic explanation methods designed for brain networks. Usually the brain is divided in well established regions of interest (ROI) that act as vertices and a link between two ROIs is assumed when both are co-activated. In this domain, the neuroscientists have many tools for providing diagnostics so automatic classification is not always very interesting for them. However, the counterfactual examples are highly interesting for the neuroscientist as they can provide new insights and hypothesis, about the employed models [2].

The proposed approach is based on edge perturbations, allowing to add and remove edges from the graph to generate a counterfactual example. In the general case the search space for this problem has size $|G(V)|= 2^{\frac{n(n-1)}{2}}$, thus an exhaustive search is not feasible in practice. The common logic of OBS and DBS is to use an heuristic bidirectional search. In the first stage, the method is going to perform perturbations to the edges of the original graph G until a counterfactual graph G' such that $\Phi(G) \neq \Phi(G')$ is obtained. In the second stage the method is going to rollback some of the perturbations made in the first stage, so to decrease the distance between G and G' while maintaining that $\Phi(G) \neq \Phi(G')$. The main difference between OBS and DBS is that OBS chooses the edges to perturb randomly and thus does not require any previous knowledge about the dataset. On the other hand, DBS learns from the dataset which edges are more common in each class of graph and uses this value to decide the order for perturbing the edges.

1. *Model agnosticism and model access*: After each perturbation, to the original graph or the counterfactual example, OBS and DBS feed the new graph to the Oracle and check if the decision has changed. Both explanation methods consider the oracle a black box and do not require to access to its internal information. Thanks to this approach OBS and DBS are model agnostic and can be used with any oracle.
2. *Factual-based explanations and minimal CE*: OBS and DBS are specifically designed to provide counterfactual explanations and doesn't rely on any existing factual explanation method. Furthermore, both methods are focus on the problem of finding minimal GCE (see equation 5). To achieve this, in the second stage, these explanation methods try to minimise the distance between the original graph and the counterfactual example. However, given the heuristic nature of these explainers, it is not always possible to generate a minimal GCE.
3. *Domain agnosticism and training data accessibility*: In general, both explanation methods are general enough to be applied to different domains. However, the implementation of the explainers made some assumptions based on the nature of the brain networks domain. The first assumption is that the prediction task is a binary classification problem, while the second one requires that all the graphs in the datasets contain the same vertices and that the identifier for each vertex is consistent across the entire dataset. The former limitation is not difficult to overcome with some small modifications to the implementation, while the later cannot be overcome without making significant changes to the algorithm and affecting its performance due to the need of performing graph matching when calculating graph edit distance between different instances. OBS does not require access to any additional data beyond the input instance for which it is going to generate a counterfactual example. On the opposite, DBS requires a trained datasets with labeled examples, so it can learn which edges appear more frequently in the instances belonging to each class.
4. *Classification task and generation type*: OBS and DBS are designed for providing explanations to the decision of oracles performing graph classification tasks. As previously mentioned, the explainers employ edge addition and removal operations in order to produce a counterfactual example. One particularity of these methods is

they receive a bound, to the number of calls to the oracle, as a parameter. The reason is that the prediction process can be computationally expensive and could be desirable to bound the number of calls to the oracle, even if it means obtaining not minimum GCEs.

Molecular Explanation Generator (MEG) **[93, 92]** is a method designed to explain molecular graphs. The approach uses a Reinforcement Learning (RL) agent to generate counterfactual examples given an input molecule. The reward function of the agent binds together Equation 12 with a term regulating the change in prediction scores, which is inherently task-dependent. For the regularisation term the authors explored three different approaches. The first one consisting in using Tanimoto similarity over the binary Morgan fingerprints [108]. The second approach used cosine similarity in the vector representation of the molecules used by the oracle. Finally, the third approach was a convex combination of the two previous ones. To achieve validity of the counterfactuals, the authors based the implementation of the generator on the MolDQN [163] model. In general each time the RL counterfactual generator produces an instance it is verified that it is a valid molecule, and it is discarded if it is not.

1. *Model agnosticism and model access*: MEG requires access to the molecules vector representation used by the oracle to calculate the similarity between instances. This requirement prevents the method to be completely model agnostic, and by default is assumed than the oracle is a GNN. However, the explainer could be adapted to work with other oracles as long as they provide access to a vector embedding of the graphs.
2. *Factual-based explanations and minimal CE*: The explainer is especifically designed for producing GCE and it is not based on existing factual explanation techniques. From the definition of GCE, provided by the authors in Equation 12, it is clear that MEG is focused on generating minimal CE. To force the explainer to produce minimal GCE, the authors included the similarity between the original graph and the counterfactual graph in the reward function of the RL agent.
3. *Domain agnosticism and training data accessibility*: MEG is specifically designed for the molecular domain, and enforces that the graphs are valid molecules. The explainer uses domain knowledge to provide more meaningful molecular counterfactual explanations, but at the same time loses its capacity to be generalised to other domains. The RL GCE generator could improve with the use of training data, but does not require it because it will learn each time its used to provide a new explanation.
4. *Classification task and generation type*: In the molecular domain, the problems involving vertex and edge classification are less common than those involving graph classification tasks. MEG mainly provide explanations for oracles focusing on the graph classification task. However, the method can be also used to explain the decisions in regression tasks. In general MEG is one of the few methods that employs additions and removals of both vertices and edges in its generation process.

Counterfactual and Factual ***(CF^2) [124]*** is a method designed to produce factual GNN explanations by balancing factual and counterfactual reasoning. The GNN explanation problem is presented as a multi-objective optimisation problem where the generated counterfactual should comply with specific properties. First, the factual property requires the explanation graph to be a subgraph of the input instance such that the prediction for the input graph and the explanation one is the same. Second, the counterfactual property requires that once the explanation subgraph is removed from the input instance, the prediction on this new graph differs from the original. Finally, the produced explanation should be simple, so the smaller the explanation size the better. Despite being a factual method, we included CF^2 because due to the counterfactual property of its explanations, the graph resulting from subtracting the explanation graph from the input instance can be considered a counterfactual explanation for that instance.

1. *Model agnosticism and model access*: The method does not require access to the oracle internal representation. The explainer can be used with different kinds of oracles as it only needs access to the prediction.
2. *Factual-based explanations and minimal CE*: As mentioned before, CF^2 is a perturbation-based graph factual method. However, it requires the explanations to comply with the counterfactual property, that means removing the explanation subgraph from the original graph will produce a change in the prediction. The main issue with the factual-based approach is that it is optimised to produce small factual explanations, but in general does not produce minimal counterfactual explanations, as these are usually just a subgraph of the factual explanation. Moreover, the method solely performs subtraction operations on the original graph.
3. *Domain agnosticism and training data accessibility*: The authors of CF^2 provide experimental results demonstrating the effectiveness of their method on synthetic, citation, and molecular datasets. In this way, CF^2 is adaptable to any domain as long as it encompasses graph data. Additionally, the explainer does not require labelled data to produce a counterfactual example since the explanation produced is a mere derivation of the removal of the factual subgraph from the original instance.

4. *Classification task and generation type*: The method can be used for explaining the predictions of oracles focused on both graph classification and vertex classification tasks. To generate the explanation CF^2 performs edges, vertex, and vertex attribute removals on the original graph instance. A limitation of this approach is that edge and vertex additions are not considered because the factual explanations are a subset of the input graph, while, in general, counterfactual explanations are not.

4 Evaluation

A fundamental aspect of successful research is to provide evidence of the effectiveness of the proposed solution. Evidence is carried out through a systematic inquiry to assess an object, program, practice, activity, system or method. This assessments, in the data science domain as in the GCE one, is typically conducted following quantitative evaluations which include standardised tests and a variety of measurements following a fixed experimental protocol. The evaluation protocol for GCE methods comprises the following steps: *i)* the researcher train the classifier on a chosen dataset, *ii)* the explainer, that must be evaluated, takes in input the trained model and a chosen instance and produces an explanation; *iii)* the produced explanation and the traces of runtime of the explainer are evaluated accordingly to several evaluation measures. The process can be repeated for several datasets, explainers and classifiers and the resulting measures are typically aggregated in tables and plots.

For this reason, here, we first present (see Section 4.1) the datasets typically employed in the literature for the evaluation of GCE methods. Then we discuss the metrics (see Section 4.2) useful to assess the performances of the counterfactual explainers by providing an intuition about the benefits and drawbacks of each ones. In Section 4.3, we shed light on the evaluation strategies followed by the works discussed in this survey. In addition, in Section 4.1.1 we depict the process of generating synthetic datasets that future researchers can refer to in order to expand their experimental scenarios.

4.1 Datasets adopted in the literature

Due to the absence of standardised and well-established benchmarks in the literature, it is difficult to compare the approaches presented in Section 3. The surveyed studies typically compare themselves with simple baselines rather than state-of-the-art solutions by adopting a heterogeneous set of synthetically generated datasets and ad-hoc datasets that are not part of an established benchmark. To bridge this gap for future researchers, we provide Table 4 as an index of available datasets. Thus, researchers can adopt them to evaluate and compare their performances with those proposed in the literature. In Section 4.1.1, we discuss the generating strategies of the synthetic datasets and of their importance, and, in Section 4.1.2, we discuss real-world datasets adopted in the literature.

4.1.1 Synthetic datasets

Synthetic datasets represent a well-structured scenario w.r.t. real-world datasets because the properties of the underlying graph have been generated by a well-defined mechanism[17]. The advantage of having synthetic graphs is thus that the performances evaluated on them can be easily understandable due to the fact that the properties of the datasets are well known and can be easily controlled and modified. Moreover, adopting a synthetic dataset, it is possible to know which is the minimum counterfactual example for a given input instance and compute the distance (minimal) between the two. For example, supposing a classification task that classifies graphs between cyclic or acyclic, if we have a tree graph as input, then the minimal counterfactual example[18] can be obtained by adding a single edge between any two vertices of the graph. Hereafter, we first summarise the meta approach that we derived from [150], and then we present the synthetic datasets with their strength.

The generative approach, presented here for readability in its binary version, can be summarised in three steps. The first one is dedicated to generating a base graph with specific characteristics (e.g. the number of vertices and edges). Typically the edges are randomly generated but can follow a certain model as the Erdős–Rényi one. The second step is dedicated to generating well-known motifs (e.g. cycle, triangle, infinity loop). Note that the second step can be skipped if the researcher wants to use a static motif (typically handcrafted) rather than a generated one. In both cases (fixed or generated motif) this step must assure that the base graph does not already contain the motif that we like to add, otherwise it might lead to an uncontrolled scenario. Finally, the chosen motifs are connected to the base graph while controlling that this step does not produce additional motifs similar to the added one. The whole dataset can be generated by including graphs generated solely by the first step (typically labelled as the 0 class) and those which follow

[17]Usually, synthetic graph datasets are generated via programmed processes that rely on specific algorithms and constraints.

[18]Recall that the minimality of a counterfactual example does not strictly depend on the explainer. Rather, the oracle who classifies the generated examples plays an important role in considering them as counterfactual or not. In cases where the explainer performs well but the oracle lacks, the literature has proposed to assess the oracle's performance by measuring its accuracy.

all three steps (labelled as the 1 class). The process can be generalised to the multi-class scenario by adopting more than one class of motif easily distinguishable and by repeating the second step according to the number of classes. To conclude, the researcher is free to choose the number of instances present in the dataset and the distribution among the chosen classes. Now we provide the reader with the description the synthetically generated graphs: **Tree-Cycles** [150]: The base structure of the dataset is an 8-level balanced binary tree. Then, 80 six-node cycle motifs are added to the instances, and attached to random vertices of the base graph. Vertices in the dataset are binary classified depending if they belong to a cycle motif or to the base tree.

This approach was generalised by [103]. The base structure of each instance of the dataset is a random tree. Then, cycle motifs are added to each instance, where their length is a user-specified parameter. The resulting graph is then binary classified if it contains a cycle or not. The user can control the number of generated instances, the number of vertices per instance, and the number of edges that connect the vertices. Figure 9 illustrates a toy example belonging to this dataset. Notice that the cycle motif added to the tree (left-side) are attached to a single vertex to avoid creating more cycles than needed. In other words, if the cycle were attached to two nodes, three different cycles would be created instead of one. This generation method follows in spirit what the authors in [150] have proposed, with the difference of taking care of providing minimal counterfactual explainability.

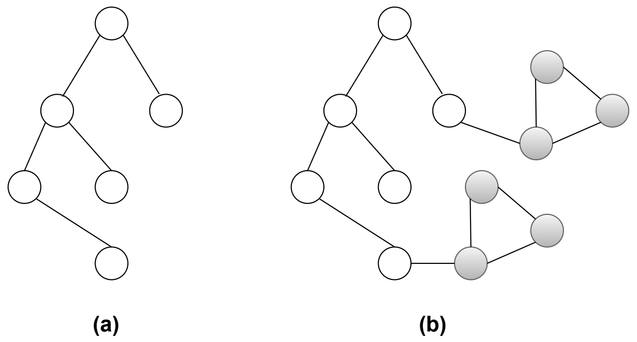


Figure 9: Example of Tree-Cycles dataset instances. Instance a belongs to class 0 while instance (b) belongs to class 1

Tree-Grid [150]: Is a node classification dataset that follows the same idea as Tree-Cycles, except that 3-by-3 grid motifs are attached to the base tree graph in place of the cycle motif.

Tree-Infinity [103]: It follows the same approach of the Tree-Cycles dataset, but instead of cycles, there is an infinity shape attached to the base graph. Thus, if cycles in the base graph are allowed but if there is no infinity shape, its class will be *no-infinity* (class 0). The number of vertices in the infinity pattern can be user specified with the constraint for it to be an odd number because the subgraph on the left and on the right of the infinity shape are connected via a single vertex.

The authors proposes the Tree-Infinity because oracles, in Tree-Cycle, can learn how to count vertices and predict whether the input is cyclical or not.

BA-Shapes [150]: Is a dataset designed for node classification. In this dataset, each instance is created by starting with a base Barabási-Albert (BA) graph on 300 nodes and a set of 80 five-node "house"- structured network motifs, which are attached to randomly selected nodes of the base graph. The resulting graph is further perturbed by adding $0.1 \cdot |V|$ random edges. In a house-structured motif, the nodes can be in 3 different areas: the top, the middle and the bottom of the house. Therefore there are 4 different classes, corresponding to them plus a class for the nodes that do not belong to the house.

BA-2motifs [77]: Is a dataset for graph classification containing 800 graphs. Instances use BA graphs as base graphs. Half of the graphs in the dataset are attached with "house" motifs and the rest are attached with five-node cycle motifs. Graphs are binary classified according to the type of attached motifs.

BA-Community [150]: Is a dataset for vertex classification, where the graph is formed by the union of two BA-Shapes graphs. Two Gaussian distributions are utilised to sample node features, one for each BA-Shapes graph. Vertices are assigned to one of 8 classes based on their structural roles and community memberships.

4.1.2 Real datasets

Besides the synthetic datasets provided above, the works in the literature have relied also on real datasets. on the following: The real datasets can be divided into four domains: -omics, molecular, social and text data.

ASD [2]: Autism Spectrum Disorder (ASD) is a graph classification dataset taken from the Autism Brain Imagine Data Exchange (ABIDE) [28]. In particular, it is focused on the portion of the dataset containing children below nine years of age [68]. It contains 49 individuals in the condition group, labelled as Autism Spectrum Disorder (ASD), and 52 individuals in the control group, labelled as Typically Developed (TD). The data is obtained by means of functional magnetic resonance imaging (fMRI). Each instance is a graph where the vertices represent brain Regions of Interest (ROI) and an edge represents co-activation between two ROIs.

ADHD [2]: Attention Deficit Hyperactivity Disorder (ADHD) is a graph classification dataset, taken from the USC Multimodal Connectivity Database (USCD) [19], in particular from the $ADHD200_CC200$ study. The data is obtained by means of functional magnetic resonance imaging (fMRI). Each instance is a graph where the vertices represent brain ROIs and an edge represents co-activation between two regions. The dataset contains 190 individuals in the condition group, labeled as ADHD and 330 individuals in the control group, labeled as TD.

BBBP [141]: Blood-Brain Barrier Permeation is a molecular dataset for graph classification. Predicting if a molecule can permeate the blood-brain barrier is a classic problem in computational chemistry. The most used dataset comes from Martins et al.[83]. The authors, compiled a data set of 2053 molecules selected from a number of publications discussing BBB penetration. Of these, only 1970 were used for modeling purposes as all compounds that exceeded a molecular weight of $600Da$ were excluded. Most studies divide molecules as being able or not to cross the BBB so the graphs in the dataset were binary classified accordingly. In total there are 1570 instances capable of crossing the Blood-Brain Barrier and 483 not capable of it.

HIV Activity Prediction [141]: Is a molecular dataset for graph classification. The National Institute of Allergy and Infectious Diseases has made a systemic study of compounds that can inhibit HIV resulting in large compound datasets that classifies compounds based on their ability to inhibit HIV. This dataset was prepared by the Drug Therapeutics Program (DTP) for AIDS antiviral screening for more than $40,000$ compounds [30, 35], and used in a Kaggle competition.

Mutagenicity [57]: Is a molecular dataset for graph classification. Mutagenicity is one of the numerous adverse properties of a compound that hampers its potential to become a marketable drug [57]. Toxic properties can often be related to chemical structure, more specifically, to particular substructures, which are generally identified as toxicophore. The dataset contains $4,337$ molecules binary classified into mutagenic or non-mutagenic according to the presence of these substructures. The average number of vertices and edges per instance are 30.32 and 30.77 respectively.

NCI1 [138]: Is a molecular dataset for graph classification. It contains data published by the US National Institute for Environmental Health Sciences and consists of bio-assays of different chemical compounds on rodents to study the carcinogenicity properties of the compounds [138]. The dataset contains $4,110$ chemical compounds which are categorised as either positive or negative to cell lung cancer. The average number of vertices and edges per instance are 29.87 and 32.30 respectively.

TOX21 [58]: Is a collection of molecular datasets for graph classification. More than 30 percent of promising pharmaceuticals have failed in human clinical trials because they are determined to be toxic despite promising pre-clinical studies in animal models [62]. Creating new methods for assessing chemical toxicity has the potential to improve how scientists evaluate environmental chemicals and develop new medicines. The Tox21 consortium created a dataset of $10,000 compounds$ including environmental chemicals and approved drugs and how they will affect human health and the environment. Tox21 is divided into multiple datasets, each corresponding to a particular assay. Each instance from the datasets is binary classified into toxic or non-toxic.

ESOL [147]: Is a molecular dataset for regression tasks on graphs. It is a small dataset consisting of water solubility data for 1128 compounds. The dataset has been used to train models that estimate solubility directly from chemical structures (as encoded in SMILES strings). These structures don't include 3D coordinates, since solubility is a property of a molecule and not of its particular conformers. The file containing the dataset is named *"delaney-processed.csv"* in the github repository whose link is provided in table 4.

CiteSeer [40]: Is a citation network for node classification. The dataset consists of 3312 scientific publications (vertices) classified into one of six classes, and of 4732 links indicating a citation from one publication to the other. Each publication in the dataset is described by a 0/1-valued word vector indicating the absence/presence of the corresponding word from the dictionary. The dictionary consists of 3703 unique words.

MIMIC-III-50 [88] is an open-access dataset of text and structured Electronic Medical Records (EMR) from a hospital ICU. Each admission is tagged by human coders with a set of $ICD-9$ codes, describing both diagnoses and procedures which occurred during the patient's stay. There are 8921 unique $ICD-9$ codes present in the dataset, including 6918 diagnosis codes and 2003 procedure codes. MIMIC-III-50 consists of the records labelled with at least one of the 50 more frequent labels. The result is 8067 records for training, 1574 for validation, and 1730 for testing. Should be noted that instances in this dataset are not graphs, vertices, or edges, but EMRs instead. For this reason this dataset is only used by CMGE [145], that is specifically designed for this domain.

Table 4: The table shows the datasets used in the literature reporting their domains, the link to their repository, and the papers that employed them for the evaluation: RCExplainer[11], CF^2[124], CF-GNNExplainer[75], BOS and BDS[2], MACCS[141], MEG[93], CMGE[145], GRETEL[103].

Dataset	Domain	Publicly Available Repository (Data or Code)	Employed by
Tree-Cycles [150]	synthetic	`https://github.com/RexYing/gnn-model-explainer`	[11, 124, 75]
Tree-Grid [150]	synthetic	`https://github.com/RexYing/gnn-model-explainer`	[11, 75]
Tree-Infinity	synthetic	`https://github.com/MarioTheOne/GRETEL`	[103]
BA-Shapes [150]	synthetic	`https://github.com/RexYing/gnn-model-explainer`	[11, 124, 75]
BA-Community [150]	synthetic	`https://github.com/RexYing/gnn-model-explainer`	[11]
BA-2motifs [77]	synthetic	`https://github.com/flyingdoog/PGExplainer`	[11, 124]
ADHD [19]	-omics	`https://github.com/MarioTheOne/GRETEL/tree/main/data/datasets/adhd`	[2]
ASD [28, 68]	-omics	`https://github.com/MarioTheOne/GRETEL/tree/main/data/datasets/autism/asd`	[2]
BBBP [83]	molecular	`https://www.kaggle.com/datasets/mmelahi/cheminformatics?select=bbbp.zip`	[141]
HIV [30, 35]	molecular	`https://www.kaggle.com/datasets/mmelahi/cheminformatics?select=hiv.zip`	[141]
Mutagenicity [57]	molecular	`https://ls11-www.cs.tu-dortmund.de/people/morris/graphkerneldatasets/Mutagenicity.zip`	[11, 124]
NCI1 [138]	molecular	`https://ls11-www.cs.tu-dortmund.de/people/morris/graphkerneldatasets/NCI1.zip`	[11, 124]
TOX21 [58]	molecular	`https://tripod.nih.gov/tox21/challenge/data.jsp`	[93]
ESOL [147]	molecular	`https://github.com/deepchem/deepchem`	[93, 124]
CiteSeer [40]	social	`https://linqs.org/datasets/`	[124]
MIMIC-III-50 [88]	text	`https://mimic.mit.edu/` (Access Authorisation Request Necessary)	[145]

4.2 Evaluation Metrics

Evaluating the performance of a GCE method is a complex task given the multiple factors that can influence the quality of explanations. Hence, different metrics can be used to understand how an explainer performs and, more importantly, why the explainer generated specific counterfactual examples. Table 5 presents an overview of the used metrics in the literature. Despite measuring a complex relationship between oracle predictions and the correctness of the explanation, in practice, it is better to adopt many different simple metrics instead of complex ones - i.e. fidelity - as sometimes a single value is not enough to fully understand the behaviour of the explainer.

Table 5: Evaluation metrics used in the literature. We do not include CMGE [145] in this Table because it does not assess the performances of the explainer nor the quality of the counterfactual examples per se.

Method	Minimality Evaluation	Fidelity	Accuracy	Sparsity	Oracle Calls	Runtime	Robustness	Prob Necessity
CMGE [145]	×	×	×	×	×	×	×	×
RCExplainer [11]	×	✓	×	×	×	✓	✓	×
CF-GNNExplainer [75]	Explanation size	✓	✓	✓	×	×	×	×
MEG [93]	MEG custom similarity	×	×	×	×	×	×	×
MACCS [141]	Tanimoto similarity	×	×	×	×	×	×	×
CF^2 [124]	×	×	✓	×	×	×	×	✓
BOS & BDS [2]	Graph edit distance	×	×	×	✓	×	×	×

Runtime (t) - It measures the seconds taken by the explainer to produce the counterfactual example. The runtime provides an efficient way of measuring the efficiency of an explainer in generating counterfactual examples. Additionally, the runtime measures the entire time from when the input enters the pipeline and when the counterfactual examples are generated from the explainer considering also the execution of the oracle. Therefore, taken in isolation, the runtime can discriminate a *"good"* explainer based on the sluggishness of its oracle.

Oracle Calls [2] - Given that explainers should be model-agnostic and the computational cost of making a prediction can be unknown, it is a desirable property that the explainers perform as few calls to the oracle as possible. Given a graph G, the number of calls to the oracle is the number of times the explainer needs to enquire the oracle Φ to produce a counterfactual example G'. This metric is useful also to evaluate the number of interactions between the explainer and the oracle in a distributed system where latency and throughput are critical constraints. In this way, the fewer the interactions, the stabler the network of the distributed system is.

Correctness/Validity [44, 103] - While the previously defined metrics help to measure the performances of the explainer, there are other metrics more focused on measuring the quality of the explanations. The most intuitive quality metric is the correctness that indicates whether the explainer is capable of producing a valid counterfactual explanation (i.e. the example has a different classification from the original instance). More formally, given the original graph G, the produced example G', and the oracle Φ, the correctness is an indicator function $\mathbb{1}[\Phi(G) \neq \Phi(G')]$ defined as follows:

$$\mathbb{1}[\Phi(G) \neq \Phi(G')] = \begin{cases} 1 & \text{if } \Phi(G) \neq \Phi(G') \\ 0 & \text{otherwise} \end{cases} \tag{15}$$

Notice that this metric has not been used in the literature, but we suggest future researchers rely on it because it measures the capability of the explainer to produce correct/valid a counterfactual example - not necessarily a minimal one.

Probability of Necessity (P_n) [124] - It is a metric similar to correctness, but it is designed for explanation methods that produce multiple explanations for a given input instance G. This metric measures how many of the generated explanations for a given instance are correct. In other words, P_n is the number of explanations $G' \in \mathcal{G}'$ such that $\Phi(G') \neq \Phi(G)$ and is defined as follows:

$$P_n = \frac{\sum_{G' \in \mathcal{G}'} \mathbb{1}[\Phi(G') \neq \Phi(G)]}{|\mathcal{G}'|} \tag{16}$$

Notice that Correctness and P_n are both needed because it is possible to have $P_n = 0.99$ and still have as the returned explanation one that is not a correct counterfactual example. Contrarily, it is also possible to have correctness of 1 as far as the best explanation is a proper counterfactual example, even if most of the computed explanations are not counterfactuals.

Sparsity (Λ) [153] - It measures the similarity between the input graph and the counterfactual example produced in proportion to the attributes of the input instance. Recall that the attributes of an instance x, can be related to:

- vertices $a(v)$ if x is a vertex;
- edges $\bar{a}(e)$ if x is an edge;
- or the entire graph $\tilde{a}(G)$ if x is a graph.

More formally, let $|x|$ be the number of attributes, as described above, of the input instance x, x' the counterfactual example generated, and $\mathcal{D}_{inst}(x, x')$ the distance between x and x'. Sparsity is then defined as follows:

$$\Lambda(x, x') = 1 - \frac{\mathcal{D}_{inst}(x, x')}{|x|} \tag{17}$$

Oracle Accuracy [37] - Even if the explanation methods are model agnostic, the quality of the predictions can have a big impact in the quality of the explanations. The reason behind this is that the explainer is providing an explanation about the behaviour of the model. If the predictions of the model are not accurate, then the provided explanations can have no sense for the user. Therefore, given the input graph G, its ground truth label y_G, and an oracle Φ, the accuracy is defined as $\chi(G, G') = \mathbb{1}[\Phi(G) = y_G]$.

Fidelity (Ψ) [153] - Given the relation between the oracle performance and the explanation performance, fidelity measures how much the explanations are faithful to the oracle, considering its correctness. Given the input graph G, its ground truth label y_G, and the counterfactual G', fidelity is defined as follows:

$$\Psi(G, G') = \chi(G, G') - \mathbb{1}[\Phi(G') = y_G] \tag{18}$$

Notice that $\Psi(G, G')$ can assume three values as follows: textbf1 entails that both the explainer and oracle are working correctly. It is trivial to verify that the first indicator need to produce a 1 (i.e. $\Phi(G) = y_G$) and the second indicator a 0 (i.e. $\Phi(G') \neq y_G$). **0** and **-1** describes that something is wrong with either the explainer or the oracle. However, we cannot attribute the incorrect function to the explainer or the oracle. This is a shortcoming of fidelity since it bases the assessment of the correctness $\mathbb{1}[\Phi(G') = y_G]$ on the ground truth label y_G instead of the prediction $\Phi(G)$ as illustrated in Equation 15.

Robustness [11] - This metric quantifies how much an explanation changes after adding noise to the input graph. For an input graph G and the explanation G', a perturbed graph G_p is produced by adding random noise to the node features of G and randomly adding or deleting edges, while keeping that $\Phi(G) = \Phi(G_p)$. Then the change in the explanations G' and G'_p is measured. Bajaj et al. [11] Consider the top-k edges of G' as the ground truth and compare G'_p against them. Then they compute a receiver operating characteristic (ROC) curve and evaluate the robustness by the area under curve (AUC) of the ROC curve. However, they approach considers the explainer assigns an importance score to the edges of the graph. In general it is possible to use different functions to calculate the similarity between G' and G'_p.

Minimality Evaluation As provided in Definition 5, it is important to study the minimality of the counterfactual examples. Notice that, to provide a minimal counterfactual example, one needs to define metrics of similarity/distances between the generated counterfactual G' and the input graph G. Some of the works in Table 5 rely on already-established similarity/distance measures. Meanwhile, MEG [93] exploit a custom similarity measurement which is based on the Tanimoto similarity. Here we report each of these metrics and provide comments accordingly:

- *Graph Edit Distance (GED)* [111] - It measures the structural distance between the original graph G and the counterfactual one G'. Suppose that $p_i \in \mathcal{P}(G, G')$ is a path consisting of actions (i.e. adding/removal of a vertex/edge) to be performed on G to produce G'. Notice that there can be multiple paths of actions that transform G to G'. Each path has specific actions $r_j \in p_i$ that has a cost attached to it, $\gamma(r_j)$. Therefore, given G, G', and a set of actions $\mathcal{P}(G, G')$, the graph edit distance can be defined as follows:

$$GED = \min_{p_i \in \mathcal{P}(G,G')} \sum_{r_j \in p_i} \gamma(r_j) \tag{19}$$

In general, given two counterfactual examples we prefer the one closer to the original instance G, as it provides shorter action paths on G to change the outcome of the oracle. However, GED provides a global value, then sometimes it can be useful also to employ a relative metric that considers the size of the instance as in the case of the sparsity (see Equation 17) to measure the performance of the algorithm over multiple instances.

- *Explanation Size* [75] - Is the difference between the original graph G and the counterfactual G' explanation. The idea is very similar to that of Graph Edit Distance, and it considers counterfactual explanation as a set of edit actions $\mathcal{P}(G, G') = \{p1, \ldots, p_n\}$ to perform on G in order to transform it into G'. Then the explanation size can be defined as $|\mathcal{P}(G, G')|$. Lucic et al. [75] consider it the number of edge removals necessary to transform G into G'.

- *Tanimoto Similarity* [12] - It is used to calculate the similarity between two molecules represented as binary vectors and it is one of the most used similarity functions in the chemical domain. Being the original molecular graph G and the produced explanation G' then the Tanimoto Similarity can be defined as follows:

$$\tau(G, G') = \frac{\sum_{j=1}^{n} b(G, j) \cdot b(G', j)}{\sum_{j=1}^{n} b(G, j)^2 + \sum_{j=1}^{n} b(G', j)^2 - \sum_{j=1}^{n} b(G, j) \cdot b(G', j)} \tag{20}$$

where $b(G, j)$ is a function that, given a molecular graph G and a integer index j, returns the value of the bit with the corresponding index in the binary representation of the graph.

- *MEG Similarity* [93] - Is a convex combination of the Tanimoto similarity (structural) and the neural encoding similarity of the model. The neural encoding similarity is a cosine similarity on the embedding representation of the graphs. Numeroso et al. [93] show that this metric obtains better results than its component functions. Given two graphs G and G' MEG similarity can be defined as follows:

$$S_{MEG} = \alpha_1 \tau(G, G') + \alpha_2 \cos(G, G') \quad | \quad \sum_i \alpha_i = 1 \tag{21}$$

4.3 Evaluation strategies for GCE in the Literature

In this section we report the evaluation strategies performed by any of the works included in this survey:

- **DCE** [36]: Distribution Compliant Explanations, firstly introduced in [36], refers to explanations that belong to the same data distribution as the input instance. In particular, DCE is firstly used as a baseline in the literature in [2]. It does not make any assumption about the underlying dataset and searches for a counterfactual instance in it instead of generating a new counterfactual. In particular it searches for a graph such that:

$$\arg \min_{G' \in \mathcal{G},\ \Phi(G) \neq \Phi(G')} \mathcal{D}_{inst}(G, G') \tag{22}$$

- **CMGE** [145]: the authors evaluate their method mainly in the factual explainability. To do so the use their method as a feature extraction technique for Electronic Medical Records (EMR) showing the obtained accuracy using the identified features: they extract the 3 main factual features identified by the method and submit them for human evaluation. The main employed data set is composed of EMR instances in Chinese language, for lymphedema patients. The dataset is custom built and it is not publicly available. The authors also used the English EMR benchmark "MIMIC-III-50" [88] for the task of assigning multiple ICD codes to EMRs. The proposed method is mainly tested for the link prediction task and its explanation capabilities are not directly evaluated or compared against other counterfactual explanation methods.

- **CF-GNNExplainer** [75]: Due to the fact that the counterfactual generator only removes edges from the input instance, the authors choosed to compare it against factual GNN explanation methods. The four baseline factual explainers used in their comparison include: Random, Only-1hop, Rm-1hop, and GNNExplainer [150]. The random perturbation was meant as a sanity check, and basically implies randomly removing edges of the graph and keep track of the minimal perturbation that produced a counterfactual. Only-1hop and Rm-1hop baselines are based on the 1-hop neighbourhood of the vertex (i.e., its ego graph). Only-1hop keeps all edges in the ego graph of the vertex while Rm-1hop removes all edges in the ego graph of the vertex.

They use three synthetic datasets namely Tree-Cycles, Tree-Grids, and BA-Shapes for the node classification task. , They measures the performances of the different explanation methods accordingly to the *Fidelity*, *Sparsity* and *Explanation Size* metrics. The experimental results shown that for all three datasets, CF-GNNExplainer could generate counterfactual examples for the majority of nodes in the test set, while only removing a small number of edges.

- **MACCS** [141]: This method is domain-specific tied to the molecular domain. The authors does not provide any comparison to other state-of-the-art methods. Three real-world datasets were used to test MACSS, namely BBBP [83], ESOL [147], and HIV [30, 35]. To test MACCS on BBBP, the authors developed a random forest model using molecular descriptors as features. For predicting the solubility of molecules, in ESOL, a gated recurrent unit (GRU) recurrent neural network (RNN) was used. Also, A GCN was used as oracle for the HIV dataset. The Tanimoto Similarity was used to evaluate the performance of MAACS in all three datasets.

- **RCExplainer** [11]: This explanation method is based on factual explanations, so the authors conducted series of experiments to compare their method with the state-of-the-art factual GNN explanation methods including GNNExplainer [150], PGExplainer [77], PGM-Explainer [136] and CF-GNNExplainer [75]. For those explainers that identify a set of vertices as an explanation, they used those vertices to induce a subgraph of the input graph, and then used the set of edges of the induced subgraph as the explanation. On the other hand, for the methods that identify a subgraph as an explanation, they directly used the set of edges of the identified subgraph as the explanation.

The explanation performance was evaluated on two typical tasks: the graph classification task that uses a GNN to predict the class of an input graph, and the node classification task that uses a GNN to predict the class of a graph node. For the graph classification task, one synthetic dataset BA-2motifs [77], and two real-world datasets, Mutagenicity [57] and NCI1 [138] were used. On the other hand, for the node classification task, four synthetic datasets BA-Shapes, BA-Community, Tree-Cycles and Tree-Grid, reported by Ying et al. [150], were used. The *Fidelity*, *Robustness* and *Runtime* metrics were used for comparing the performance of the explanation methods.

- **OBS and DBS** [2]: This methods are designed for the brain networks domain. A simple Data Search method (DS) is provided as baseline for the comparison. The DS method searches for a counterfactual instance, with the minimal distance to the input graph, in an existing dataset. To test their proposal the authors used the Autism Spectrum Disorder (ASD) dataset taken from the Autism Brain Imagine Data Exchange (ABIDE) [28], and the Attention Deficit Hyperactivity Disorder (ADHD) dataset taken from the USC Multimodal Connectivity Database (USCD) [19]. They use the edit distance between the counterfactual example and the original instance, and the number of calls to the oracle as the evaluation measures. In order to find a ground truth, the authors use a very simple white-box classifier for which it is possible to see the inner logic. With this aim, a 2-dimensional embedding and a linear classifier where used to geometrically calculate the optimal counterfactual.

- **MEG** [93]: This method is designed to provide counterfactual explanations for molecular graphs. Given the domain-specific nature of the method, the authors did not provided any comparison with state-of-the-art methods. To test MEG two popular molecule property prediction benchmarks TOX21 [58] and ESOL [147] were used. Preliminarily, the authors scanned both datasets to filter non-valid chemical compounds. They considered structures to be valid molecules if they pass the RDKit sanitisation check. In the end, TOX21 comprises 1756 samples, equally distributed among the two classes, while ESOL includes 1129 compounds. A split of 80%/10%/10% for training, validation and test set was chosen in both datasets. During generation, MEG was employed to find 10 counterfactual explanations for each molecule in test, ranked according to its multi-objective score function. The counterfactuals were evaluated using the MEG similarity function and also using a qualitative analysis on some selected molecules.

- **Counterfactual and FactualCF^2** [124]: This method is a factual-based explanation method so the authors conducted a series of experiments to compare it with state-of-the-art factual explanation methods including: GNNExplainer [150], CF-GNNExplainer [75] and GEM [72]. The datasets used to test the method included two synthetic datasets BA-Shapes [150] and Tree-Cycles [150], as well as three real-world datasets Mutagenicity [57], NCI1 [138], and CiteSeer [40]. BA-Shapes, Tree-Cycles and CiteSeer are for node classification, while Mutagenicity and NCI1 are for graph classification. Furthermore, BA-Shapes and Tree-Cycles have ground-truth motifs for explaining the classification since they are human designed, while NCI1 and CiteSeer do not have such ground truth motifs. The used metrics include *Accuracy* and *Probability of Necessity*[19]

5 Evaluation tools for GCE

Having a systematic evaluation mechanism for explainability methods is of paramount importance for further development and extension of the GCE research field. Evaluation tools help researchers standardise their experiments and make

[19]The original paper uses also precision, recall, and F1 scores for the evaluation. Nevertheless, in this survey, we report only counterfactual-related evaluation protocols.

it feasible and easier to compare their proposed methods with the state-of-the-art leading to more consolidated claims regarding their predictions/explanations. Most of the methods enlisted in Section 3 rely on some of the metrics in Section 4.2. Nevertheless, they tend to perform their experiments on different datasets, thus hindering the replicability of their performances on other data. The surveys presented in Table 1 try to propose reproducible evaluation frameworks, respectively, SACE [44] and Extensible Counterfactual Explainers (ECE) [9]. Notice that Verma et al. [133] abstain from providing an open-source framework to solidify their contribution to the literature.

In the GCE domain the only dedicated tool which bridges the gap of the lack of extensibility and reproducibility of the evaluation frameworks proposed in the literature, is GRETEL[103]. Besides briefly describing the GRETEL framework's components in Section 5.1, we delve ourselves into discussing its main advantages according to a reproducible and extensible point-of-view (see Section 5.2), and finally, we show (in section 5.3) how to use GRETEL in practice by evaluating some Explainers and Datasets using several Metrics..

5.1 GRETEL: Design principles and core components

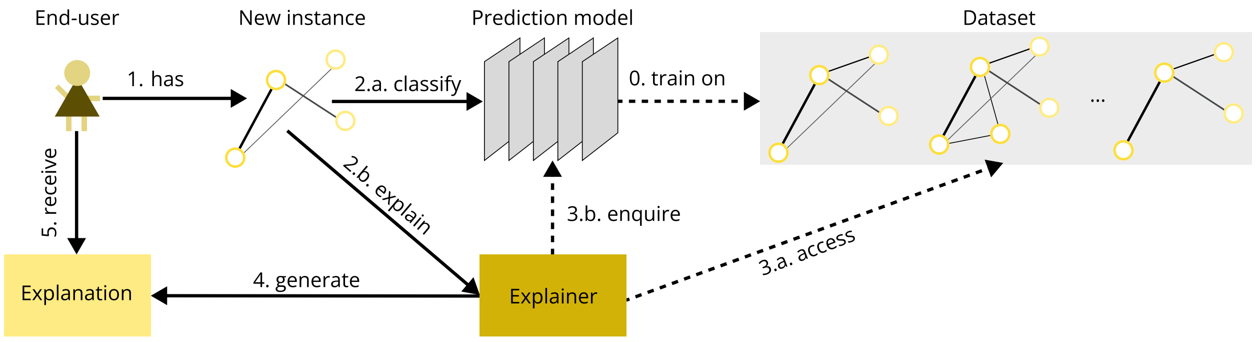


Figure 10: Typical workflow for the explanation of an instance classified by a black-box model. The dashed lines depict that the steps are optional under a privacy-preservation point-of-view. Notice that the prediction model can be trained prior to the explainability and, by loading the training weights, one can perform predictions on the new instance passed in input.

As mentioned above, GRETEL (available at `https://github.com/MarioTheOne/GRETEL`) is the only available framework that is dedicated to the evaluation of GCE methods. Hereafter we discuss its design principles and core components. According with [103] the evaluation of a GCE follow the workflow depicted in Figure 10. For visualisation purposes, we illustrate a classification scenario, but the setup is general for any generic learning task. Here, we illustrate the explanation of a new graph instance given by an end-user. Notice that, for the correct function of the explainer, we assume that the black-box prediction model has been trained on a dataset of graph data (step 0). The explainer - based on a particular post-hoc explainability method - can access the training data[20] (step 3.a) and enquire the prediction model over the outcome for the new instance (step 3.b). The new instance is passed to the prediction model (step 2.a) and the explainer (step 2.b) which generates an explanation (step 4) to pass to the user (step 5).

GRETEL follows an intuitive workflow and aids future researchers in performing exhaustive sets of evaluations. It adopts an object-oriented approach where abstract classes constitute the core of the framework. Hence, GRETEL comes with a built-in factory design pattern that helps future researchers leverage its intrinsic extensibility into forming their ad-hoc explainability scenarios. By default, GRETEL comes with its configurations, already-trained models, and the used datasets. These add-ons to the original repository allow to generate a dataset or train a model on the fly for reproducibility purposes, hence, mitigating the efforts needed to evaluate new settings.

Figure 11, shows the core components of the GRETEL Evaluation Framework, by briefly sketching them and their relationships.

For completeness purposes, here we describe GRETEL's main components referring to the structure of the original paper:

- The **Oracle** depicts a generic interface that provides extensive interactions with the employed prediction models. Besides providing basic logic to keep track of the number of calls made to the oracle, it provides the possibility to save/load the model on the fly.

[20]In privacy-preservation scenarios, the explainer does not have access to the raw training data.

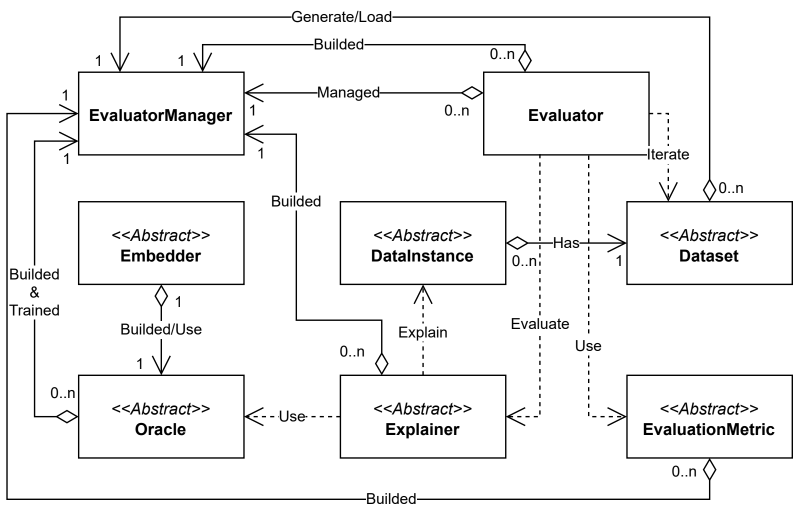


Figure 11: Overview of the main classes of the GRETEL Evaluation Framework and their relations.

- The **Explainer** represents the base class extended by ad-hoc explanation methods. Given an oracle Φ and an instance G, it provides its counterfactual explanation $\mathcal{E}_\Phi(G)$. For further experiments, the authors suggest the reader to extend this class and write their implementation accordingly.
- The **EvaluationMetric** is an abstract class useful to define specific evaluation metrics for the quality of the explainer. Again, researchers can implement their metrics and use the ones provided by default within the framework's scope.
- The **Evaluator** is responsible for carrying the evaluation of a specific explainer. Given an explainer Ξ, an oracle Φ, a set of graph instances $\mathcal{G}$, and a set of evaluation metrics $\mathcal{M}$, it collects all the performances of Ξ for each instance $G \in \mathcal{G}$ for all metrics $M \in \mathcal{M}$ according to the predictions $\Phi(G)$.

The components mentioned above constitute only part of GRETEL's ecosystem. Therefore, we refer the reader to [103] for a complete overview of the framework. Moreover, we like to note that GRETEL[21] can be extended in several ways, but the three fundamental use cases are the following:

- a data scientist wants to evaluate a brand new Explainer on the existing Datasets using an established set of Metrics;
- a company wants to evaluate, using existing Metrics, which is the best Explainer to adopt on its private Datasets thus in its business scenario;
- a data scientist or a company wants to test their new Metric on the existing Explainers and Datasets to capture unexplored aspects of the quality of the explanations;

In order to extend the GRETEL workflow the following three-step must be comprised:

1. Create a custom implementation of one of the abstract classes (e.g. Explainer, Dataset, EvaluationMetric);
2. Implement a custom factory class;
3. Use the implementation to perform explanations/evaluations;

5.2 Towards reproducibility and beyond

One of the challenges for research in machine learning is to ensure that the proposed methods and their corresponding results are sound and reliable. Specifying hypotheses, running experiments, analysing results, and drawing conclusions is the foundational process of scientific inquiry. For findings to be valid, it is essential that the experiments be repeatable and yield the results and conclusions as originally proposed [102]. While experiments in fields like biology, physics, or

[21] The source code of the framework comes with a video tutorial at `https://github.com/MarioTheOne/GRETEL/blob/main/examples/tutorial.mp4?raw=true`.

Table 6: Relationship between the data used and the analysis (including predictions and evaluations) performed in the experimental process [89]. Notice that reproducibility lies on the similar-data-similar-analysis category.

		Data	
		Similar	Different
Analysis	Similar	Reproducible	Replicable
	Different	Robust	Generalisable

sociology are made in the natural/social world, experiments in computer science are computer-designed and human-built entailing a far easier reproducibility task than the other fields enlisted. Yet, researchers have difficulties reproducing the work of others [49]. In particular, 70% of researchers failed to reproduce another paper's results in the journal Nature [13]. According to Pineau et al. [102], the reasons behind the gap of reproducible experiments in machine learning, where experiments are used to train and make predictions on observed data, are the following:

- Impossibility to access the training dataset with the same data distribution as in the original papers.
- Shortage of details in explaining the model or the training procedure.
- Unpublished code necessary to run the experiments. In some cases, the published code can also contain errors hindering the reproducibility of the experimental process in the original paper.
- Under-reporting of the evaluation metrics used to assess the performances of the prediction model.
- Improper usage of significance statistical testings to analyse results.
- Selective reporting and avoidance of adaptive overfitting.
- Overclaiming of results leading to wrong conclusions beyond the evidence presented.

Besides tackling all the above challenges for reproducibility in the literature, GRETEL provides a systematic way of using already established benchmarks and methods in graph learning. Moreover, it exploits the philosophy of object-oriented software engineering that allows users to extend certain components of the framework to contribute to the Open Science movement [134] for future research.

Reproducibility encompasses the re-execution of an experiment using the same data and the same analytical tools, including the prediction model and the same evaluation framework [126]. We invite the reader to notice the subtle difference between reproducibility and replicability [89]. In detail, replicability entails obtaining consistent results across studies aimed at answering the same scientific question each with its own data. In other words, reproducibility involves the original data and code, whereas, replicability involves new data collections and similar prediction methods used by previous studies. Inspired by [89, 102], in Table 6 we represent the relationship between the data used and the analysis performed on it. The analysis reported in the table includes the prediction model and the evaluation framework adopted. Although we are interested in reproducibility, we argue that GRETEL satisfies all the criteria depicted in Table 6. As an initial commit of the repository, GRETEL contains the implementation of some counterfactual explainers presented in Section 3.3, the majority of the datasets in 4.1 and the majority of the metrics of Section 4.2. We invite the reader to note that GRETEL is periodically maintained and updated with new state-of-the-art methods/datasets/metrics. Hence, future researchers can use the explainers and evaluation benchmarks to reproduce the experiments presented by the authors in [103]. To promote the reproducibility of these experiments, GRETEL permits and makes available the saved models and provides an on-the-fly load of their weights which can be used to make the same predictions as made in the original paper (**reproducibility**).

In particular, GRETEL follows a naturally extensible object-oriented philosophy divided into components that work as standalone modules. To this end, GRETEL permits the end-user to provide their implementations of all of its modules giving sprout to a plug-and-play scenario. Moreover, besides enlisting benchmarking datasets, the framework allows the generation of synthetic graph data based on specific patterns (e.g. cycle creation, random trees, infinity-shaped graphs). This feature aids replicating the results according to the employed metrics while maintaining the same oracles and explainers (**replicability**).

GRETEL gives the user the possibility to extend the base class of oracles and explainers while maintaining the same datasets and evaluation metrics, thus satisfying the **robustness** relationship between data and analysis. Finally, researchers can use GRETEL in terms of boilerplate code generator. In other words, they can extend the base classes of oracles and explainers for their ad-hoc scenario and generate synthetic datasets or load their own to perform original experiments (**generalisability**). More specifically, according to the dimensions in Table 6, GRETEL provides the following advantages:

- *Reproducibility* - GRETEL comes with configuration files that allow the end-user to rerun the experiments provided in the original paper.
- *Replicability* - In the configuration file, the user can add new paths pointing to other datasets and run the same experiments as in the paper. Additionally, the user can generate synthetic datasets and evaluate the performances of the explainers on them.
- *Robustness* - Besides permitting the extension of explainers/oracles, GRETEL decouples the functioning of its components from hard-coded parameters. The configuration file takes care of the correct parametrisation of the various components. Furthermore, the parameter combination produces different explainers/oracles while maintaining the same code as provided by default.
- *Generalisability* - Notice that the entire framework is extensible and personalisable to the user's explanation scenario. To aid the Open Science movement, GRETEL provides the community with Jupyter notebook tutorials on how to extend each of its components. Finally, to shrink training time consumption, GRETEL is capable of loading model weights on-the-fly if they are provided within the project's workspace, thus, enabling the community to verify the performances reported in the original work.

Taking into consideration the observations made above, GRETEL is the only GCE framework in the explainability literature that satisfies all the categories in Table 6 following compartmentalised and modular software engineering design patterns. We invite the reader to take extra care with specifying the paths to their model weight files and dataset files in their file system when trying to reproduce the experiments provided in GRETEL's online repository. Therefore, absolute path files are used instead of relative ones, which is a clear disadvantage when changing machines or even operative systems[22]. Nevertheless, the configuration files provided help avoid hard-coding the parameters of the explainers and oracles. In this way, GRETEL has the advantage to specify different parameters (e.g. graph edit distance as the minimality evaluation of the explainer). Additionally, the configuration file gives the user the possibility to specify also the strategy of generating synthetic datasets and the embedding mechanism used by the oracle. Among other advantages, this highly tailored feature of customising the experiments into bespoke evaluation scenarios provides the framework with an enhanced robust feature for future usage. Finally, the authors include tutorials[23] on how to extend each of the framework's components and provide evaluations on toy datasets with on-the-fly loaded model weights, hence contributing even more to the Open Science movement described above.

5.3 Put GRETEL into practice

To prove GRETEL's ability to evaluate explanations across diverse domains, we evaluated the explainers on a synthetic dataset (Tree-Cycles), a brain one (ASD) and a molecular one (BBBP). Table 7 reports the general statistics of the included datasets. The results in tables 8, 9 and 10 for each explainer are the average over ten runs for the non deterministic methods (standard deviation is reported in parentheses). The tables report the run time *t(s)*, the graph edit distance *GED*, the number of calls to the oracle *#Calls*, the correctness C, the sparsity Λ, the fidelity Ψ, and the oracle's accuracy *Acc*.

We tested three explainers DCE, OBS and DBS, on the Tree-Cycles dataset (see Table 8). Here, none of the explainers outperformed the others consistently. However, we must highlight that, thanks to the flexibility of GRETEL framework, for the first time we were able to evaluate OBS and DBS on other datasets besides ASD. Table 9 shows the conducted evaluations using the ASD dataset. Here, OBS and DBS outperform GED of the DCE baseline, and, more importantly, they are comparable to those obtained in [2]. This shows that the integration of OBS/DBS in GRETEL did not affect the explanation quality. However, notice that DBS performs more calls to the oracle than OBS, contrarily to what reported originally. Using the BBBP dataset, we evaluate MACCS against DCE (see Table 10). Notice that DCE and MACCS have the same order of magnitude in terms of average GED. However, because MACCS expects inputs containing a valid molecule expressed as a SMILE string, when the method is not able to provide a valid counterfactual, its correctness drops precipitously. Therefore, the correctness on BBBP is lower than the one reported in the original paper. Additionally, MACCS makes half the amount of oracle calls w.r.t. DCE making it more efficient and less time-consuming. Nevertheless, DCE still obtains good results considering that it does not leverage any domain knowledge.

[22]Absolute paths require maintenance when changing the directory of a specific file. They also require the maintainer to double-check whether the home directory has changed or not. Additionally, when changing operative systems, path separators might be different which can lead to further intricacies and tediousness in specifying them every time the running environment changes.

[23]`https://github.com/MarioTheOne/GRETEL/blob/main/examples/tutorial.mp4?raw=true`

Table 7: For each dataset it is presented the number of instances, the mean and the standard deviation of the number of vertices ($|V|$, $\sigma(|V|)$) and edges ($|E|$, $\sigma(|E|)$), and the number of instances for each class $|C_0|$ and $|C_1|$.

	$\|\mathcal{G}\|$	$\|V\|$	$\sigma(\|V\|)$	$\|E\|$	$\sigma(\|E\|)$	$\|C_0\|$	$\|C_1\|$
Tree-Cycles	500	300	0	306.95	12.48	247	253
ASD	101	116	0	655.62	7.29	52	49
BBBP	2039	24.06	10.58	25.95	11.71	479	1560

Table 8: Explainers evaluation on the Tree-Cycles dataset using the SVM oracle and the metrics described in 4.2. The values are the average over ten runs for the non deterministic methods (standard deviation is reported in parentheses).

$Exp.$	$t(s)$	GED	$\#Calls$	C	Λ	Ψ	Acc
DCE	7.47(2.61)	571.89(2.31)	501	1	0.63	0.86(0.02)	0.93
OBS	7.07(2.67)	570.04(0.3)	158.23(6.78)	0.99	0.63	0.87(0.02)	0.93
DBS	794.37(554.11)	586.11(5.62)	816.33(33.2)	0.99(0.01)	0.64	0.85(0.01)	0.93

6 Privacy, fairness, and trustworthiness in GCE

The European Union's vision of artificial intelligence poses the attention on rewarding excellence and trustworthiness of the systems, with the intent to boost research and improve industrial productivity while reinforcing safety and guaranteeing fundamental citizen rights. Having a European approach to excellence in AI [26] is important to guarantee the continent's further leading role in a worldwide economical spectrum. Moreover, the outlined Strategy aims at making the EU the centre hub for AI and ensuring that AI is human-centric while deeply exploring important aspects of fairness and trust [25].

For the aforementioned reasons, in this Section, we highlight insight into the privacy, fairness, and trustworthiness policies of AI in monitoring human-right infringements. According to the Commission[24], building trustworthy AI will create safer and more innovation-friendly digital environments for users, developers, and business stakeholders. Hence, in April 2021, the Commission proposed the first-ever legal framework aiming to provide clear requirements and obligations regarding safety provisions on AI. This particular risk-based approach proposes the following rules [27]:

- address risks specifically created by AI applications;
- propose a list of high-risk applications;
- set clear requirements for AI systems for high risk applications;
- define specific obligations for AI users and provides of high risk applications;
- propose a conformity assessment before the AI system is put into service or placed on the market;
- propose enforcement after such an AI system is placed in the market;
- propose a governance structure at European and national level.

While the EU Commission provides different layers of *risk* levels - i.e. unacceptable, high, limited, and minimal risk - they can be boiled down to *at-scale privacy infringement* blocks by automatic prediction models. Thus, the severity of the privacy infringement depends on the associated risk level. For instance, privacy violation at high-risk levels consists of identity theft, mortgage impediments via identity poisoning[25], job candidate exclusions[26], and border control breaches[27]. We invite the reader to notice that the EU Commission has placed extensive regulations for the trustworthiness of AI and the risk mitigation of its wide usage in the continent. Additionally, the regulation points the lens towards black-box models that - as depicted at the beginning of this survey (see Figure 1) - are inherently non interpretable/explainable. Hence, the predictions that allow malicious intentions, as mentioned above, can not

[24]Notice that the U.S. has not expressed any regulation throughout the entire federation. The closest policy to the one that we describe here is the California Consumer Privacy Act (CCPA) which gives users more control over the personal information that businesses can collect. Nevertheless, it does not correspond to a thorough privacy law such as the GDPR in the EU.

[25]We levy the definition of DNS poisoning [120]. Hence, we refer to identity poisoning when privacy is breached, and malicious AI systems can falsely change bank history characteristics of citizens to impede them from having a load issued.

[26]CV-sorting systems for recruitment can discriminate less-represented society groups and reject them without any further investigation of their working experience.

[27]Malicious AI can be installed in border controls can falsely alarm innocent citizens as criminals leading to their imprisonments, or even worse, let criminals infiltrate the country, thus breaching homeland security.

Table 9: Explainers evaluation on the ASD dataset using the ASD custom oracle and the metrics described in 4.2. The values are the average over ten runs for the non deterministic methods (standard deviation is reported in parentheses).

$Exp.$	$t(s)$	GED	$\#Calls$	C	Λ	Ψ	Acc
DCE	0.09(0.02)	1011.69	102	1	1.31	0.54	0.77
OBS	3.24(1.13)	9.89(0.11)	340.73(15.11)	1	0.01	0.54	0.77
DBS	83.46(34.04)	11.79(0.29)	362.05(14.56)	1	0.02	0.54	0.77

Table 10: Explainers evaluation on the BBBP dataset using the GCN oracle and the metrics described in 4.2. The values are the average over ten runs for the non deterministic methods (standard deviation is reported in parentheses).

$Exp.$	$t(s)$	GED	$\#Calls$	C	Λ	Ψ	Acc
DCE	37.51(5.21)	27.92(0.12)	2040	1	0.59	0.72	0.86
MACCS	31.35(0.97)	11.23(0.08)	1221.33(0.22)	0.4	0.19	0.23	0.86

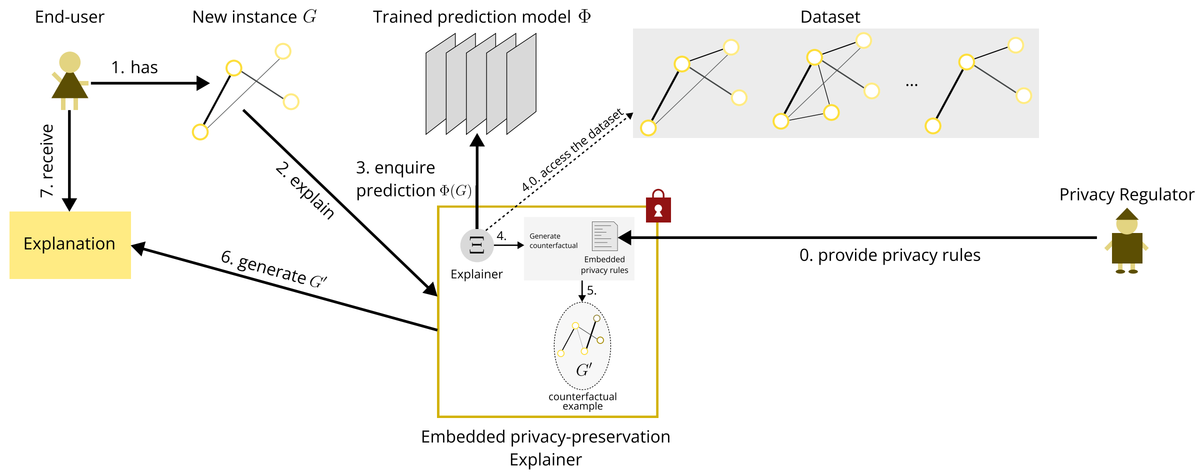
be *"deciphered"* and can not lead to a thorough understanding of what is going under the hood. According to the Commission, unacceptable- and high-risk intelligence systems are ought to be shut down completely as a move to manage privacy preservation, and be re-operative once trustworthiness conditions are met. How the Commission measures trustworthiness condition being met to render intelligent systems operative again is still to be defined, yet it poses an interesting question for future directions: i.e. *How can we measure the trustworthiness of a prediction model? Can it be quantified by a single metric as it happens with the accuracy or the mean square error?*. Contrarily, the proposal allows to use minimal-risk AI - e.g. email span filters or AI chess players - in the EU countries. These systems lack providing the end-user with an interpretation of why *an email has been marked as spam*, or why *the AI player moved the knight at cell G4*. Therefore one question that might arise is the following: *Should any non-interpretable intelligent system be considered risky regardless of the impact of its violation?* Although these questions are out of scope for this survey, we believe that interpretable (inherently, or post-hoc) models are more compliant with the EU regulation for AI than their non-interpretable counterparts. In particular, *"good"* explanations give domain non-experts the possibility to have some insight about the predictive outcome generated for a particular instance. In this regard, we delve into discussing privacy, fairness, and trustworthiness of counterfactual explanations in graphs.

Notice that counterfactual explainers can always be exploited to violate the privacy of the original data even in cases where the explainer does not have direct access to the data. By relying on the prediction of the oracle $\Phi(G)$ for a specific original instance G, the explainer Ξ can modify G in such way to obtain a new instance G' s.t. $\Phi(G) \neq \Phi(G')$ - i.e. the definition of a counterfactual example. By doing so, Ξ can pinpoint the minimal changes in the vertex/edge attributes or the structure of G (see Definition 5) to produce a new graph G' such that the predictions of both these graphs are different. Imagine that we have a social network G with users as vertices and their friendship enticed as edges between them. Without any account for the privacy of user profiles and their friendship connections, Ξ might include vertices that belong to sensitive groups - e.g. minors - thus, violating established privacy policies within the social network. Without loss of generality, privacy preservation explainers can be categorised into two classes: i.e. embedded and post-hoc privacy preservation explainers. These two classes differ substantially in how the explainer interacts with the original data and the end-user.

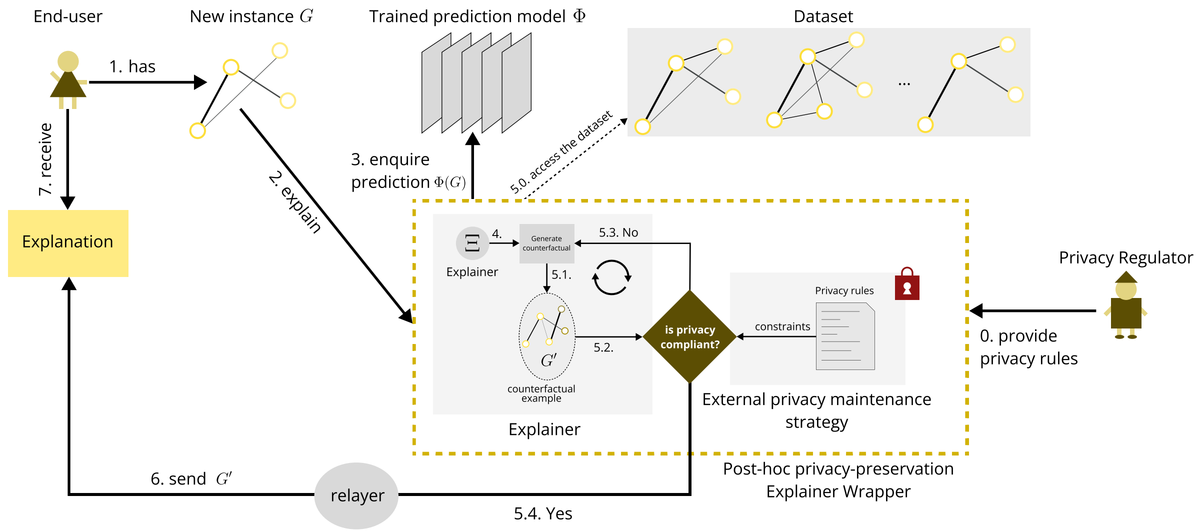
Figures 12a and 12b illustrate an embedded and post-hoc privacy-preservation counterfactual explainer strategies, respectively. Notice that in Figure 12a, the explainer has an embedded mechanism that is used to constraint the generation of the counterfactual example G' - see step 4. In this case, the explainer module needs to have full or partial access[28] to the dataset. This access is necessary when generating the counterfactual G' - see step 5 - compliant with the privacy rules fetched a priori - see step 0. Contrarily, Figure 12b illustrates how the explainer delegates the privacy compliance test to an external module that has the privacy rules sent by the regulator - see step 0. Hence, the explainer does not need to have the rules embedded within; it only requires to communicate[29] with the privacy compliance module that contains the privacy rules. Differently from an embedded privacy-preservation explainer, a post-hoc one generates an arbitrary counterfactual example G' and then uses a binary test to assess whether it complies with the privacy rules. If G' violates privacy, then the explainer repeats the generation procedure with another counterfactual in the next iteration (see steps 5.1-5.3 and notice the repeat symbol). If G' is compliant, then it gets relayed to the end-user

[28]Partial access can be considered as the access to only some attributes of the vertices/edges. For instance, consider a social network where each vertex has attributes such as age, sex, friendships, and profile pictures. A partial access policy might refrain unauthorised modules extract information about the profile pictures and the sex of users.

[29]Without loss of generality, this communication can be realised in practice by relying on APIs, where the provider is the privacy module and the consumer is the explainer module.



(a) Workflow of generating counterfactual examples G' of a graph instance G with an embedded privacy-preservation strategy. Notice that step 0 is can be done a priori to fetch specific privacy preservation rules indicated in the original data (e.g. do not use vertices corresponding to profiles of underage people).



(b) Workflow of generating counterfactual examples G' of a graph instance G with a post-hoc privacy-preservation strategy. Notice that the explainer delegates the assessment of privacy compliance to an external module. To provide a more secure interaction and isolation of the privacy rules, the explainer and the privacy maintenance strategy should be wrapped together.

Figure 12: Workflow of privacy-preserving counterfactual explainers.

(see steps 5.4, 6, and 7). It is interesting to notice that the generation process might take an enormous effort to produce a privacy-compliant counterfactual if the external privacy maintenance module limits to give a binary outcome of whether G' is compliant. Therefore, the privacy compliance test could be more complex such that it provides hints of where the failure occurred. In this way, the explainer could regulate the generation procedure to mitigate infringing the privacy in the next iteration. Notice that, in this scenario, we wrap the explainer and the external privacy maintenance strategy because, in this way, we entice the interaction of the explainer with the privacy rules without exposing them to the outside. Hence, all interactions with the external part goes through the wrapper and not the components therein. Lastly, we invite the reader to notice that both embedded and post-hoc privacy-preservation explainers can potentially access the dataset - see step 5.0 - to generate more plausible explanations. Nevertheless, if the dataset access is mandatory, then it should be handled via an encrypted and trusted channel.

Generally, fairness indicates whether the underlying black-box model learns to represent and, thus, predict the incoming instances according to some specific features leading to the so-called biased prediction. For example, let us have a toy dataset of banned users on a particular social network. Let us assume also to have the information about the race[30] of each individual in the dataset. Therefore, given an arbitrary instance, the prediction model Φ decides to output a 1 - corresponding to ban - only based on whether the person is a certain race [91] regardless of his/her connection with users that publish derogatory content or any past terms of service (ToS) violations. In this case, a produced counterfactual example under a biased-black-box predictor acts as an auditing mechanism to cope with such unfair disparities. In fact, the counterfactual example *"if the person had not been of a certain race, then he/she would not have been banned from the social network"* is unfair/biased, but it exalts the underlying unfairness of the prediction method. Hence, one can use counterfactual explainability as a monitoring strategy towards unfair predictors that are risky according to the EU Commission regulation described above. This auditing characteristic is even more critical when the discriminated group corresponds the minority group in the dataset (i.e. class imbalancement). Here, a predictor could be performing well according to a specific evaluation metric, but it remains unfair. Therefore, counterfactual examples play an important role in questioning the trustworthiness and fairness of the prediction model in any scenario. Finally, while measuring bias/unfairness of prediction model has been explored in the literature [14, 84, 100], measuring it for the counterfactual explainers remains an aspect yet to be covered. Moreover, trustworthiness of predictors can be quantified according to their performances - e.g. measuring precision, recall, or AUCPR - whereas, to the best of our knowledge, the literature in counterfactual explainability has not delved yet into measuring the fairness of the produced counterfactual examples. In any case, the fairness of the prediction method provides an insight on whether the explainer might generate biased explanations. Thus, the explainer's bias needs to be considered when evaluating the *"goodness"* of the generated explanation.

7 Open challenges and Future Works

In this Section, we provide the reader with insight on the remaining open challenges yet to be treated in the area of graph counterfactual explainability. In this way, future researchers can pinpoint the lacking aspects of the research field and concentrate in providing contributions therein.

Even though, in this survey, we extend the definition of a counterfactual explanation to the multi-class setting, more work must be conducted to clarify it in the scenario of multi-label prediction. Accordingly, the research area, to the best of our knowledge, lacks also of any method dedicated to this task.

We discussed, in Section 6, aspects of privacy, fairness, and trustworthiness in GCE. We provided a schematic intuition in Figures 12a and 12b of how privacy-preservation could be achieved by having either embedded privacy rules in the explainer or a post-hoc privacy-preservation strategy, but researchers need to focus more on defining mechanisms to preserve the users' privacy and verify its properties. To the best of our knowledge, no work proposed in GCE copes with anti-privacy-infringement mechanisms.

Similar to [82], researchers could investigate how to measure the discrimination (i.e. unfairness/bias) of the explainers. In this way, as we already mentioned, explainability can be harnessed to exalt unfairness of the underlying prediction model or to determine whether the explainer itself suffers from bias. Therefore, new generation explainers should be capable of complying with the guidelines on AI and trustworthiness which are being continuously adopted in more countries.

Besides plain graph data, counterfactual explainability can be used to provide explanations on more complex structures such as temporal graphs, and manifolds. For temporal graphs, explanations should contain a temporal component which better provide counterfactual examples for a specific time interval. Contrarily, for manifolds[31], counterfactual examples might be considered as a *surface* in the same dimensional space of the input instance, which exposes characteristics that push the prediction towards another class.

Finally, this field of research might benefit from a systematic organisation of public competitions (never done until now) regarding the evaluation of counterfactual explainers as it happens with competitions in educational data mining (e.g. KDDCup15) that encourage a uniformed evaluation benchmark with specific and well-formatted datasets.

8 Conclusion

We presented a thorough survey on the methods and best-practices for graph counterfactual explainability accompanied with rigorous formalisations of a minimal counterfactual explanation, the evaluation protocols, including datasets and measurements, and means to construct privacy-compliant explainers.

[30]We, hereby, declare to not endorse any kind of racism nor discrimination of marginalised societal groups. Any reference to race or other attributes that might be considered discriminatory are for illustration purposes only.

[31]Usually manifolds are represented as meshes of triangles in a three dimensional space.

In detail, we provided the reader with an organisation of the literature according to a uniform formal notation, thus, simplifying potential comparisons w.r.t to the method advantages and disadvantages. We emphasise that this is the first work to propose a formalisation of GCE under a multi-class prediction problem. Additionally, we provided a definition which encompasses the global minimal - i.e. the least distant counterfactual w.r.t. the original graph among all classes - for a chosen black-box prediction model.

We proposed a classification of the existing methods according to eight dimensions which aid the reader identify those methods that better suit their scenario of explainability. Besides this classification, we summarised the strengths and weaknesses of the surveyed methods. We also delve into shedding light on the benefits of a standardised evaluation protocol where we enlist synthetic and real datasets used in the literature, the adopted evaluation measures and the evaluation strategies of the surveyed methods.

Additionally, we argue that a fully-extensible and reproducible GCE evaluation framework is of paramount importance. Therefore, we illustrate GRETEL which is the first evaluation framework that concentrates in providing a highly modular architecture that permits the reader to plug-and-play with their ad-hoc explainer models, synthetic dataset generation, and evaluation metrics.

Finally, we treat privacy, fairness, and trustworthiness of explainability that are necessary to comply with AI regulations being stabilised in more countries worldwide. As a conclusion remark, we leave the reader with future directions and open challenges to be tackled in the future of this research area.

Acknowledgment

This work is partially supported by Territori Aperti a project funded by Fondo Territori Lavoro e Conoscenza CGIL CISL UIL (`https://territoriaperti.univaq.it/`), by the European Community H2020 programme under the funding schemes INFRAIA-01-2018-2019 Res. Infr. G.A. 871042 SoBigData++ (`http://sobigdata.eu/`), XAI: Science and technology for the eXplanation of AI decision - ERC Advanced Grant 2018 G.A. 834756 and by the HPC & Big Data Laboratory of DISIM, University of L'Aquila (`https://www.disim.univaq.it/`).

References

[1] Agnar Aamodt and Enric Plaza. Case-based reasoning: Foundational issues, methodological variations, and system approaches. AI communications, 7(1):39–59, 1994.

[2] Carlo Abrate and Francesco Bonchi. Counterfactual graphs for explainable classification of brain networks. In Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining, pages 2495–2504, 2021.

[3] Michael Affenzeller, Bogdan Burlacu, Viktoria Dorfer, Sebastian Dorl, Gerhard Halmerbauer, Tilman Königswieser, Michael Kommenda, Julia Vetter, and Stephan Winkler. White box vs. black box modeling: On the performance of deep learning, random forests, and symbolic regression in solving regression problems. In International Conference on Computer Aided Systems Theory, pages 288–295. Springer, 2019.

[4] Sajad Ahmadian, Milad Ahmadian, and Mahdi Jalili. A deep learning based trust-and tag-aware recommender system. Neurocomputing, 488:557–571, 2022.

[5] Guillaume Alain and Yoshua Bengio. Understanding intermediate layers using linear classifier probes. arXiv preprint arXiv:1610.01644, 2016.

[6] Kenza Amara, Rex Ying, Zitao Zhang, Zhihao Han, Yinan Shan, Ulrik Brandes, Sebastian Schemm, and Ce Zhang. Graphframex: Towards systematic evaluation of explainability methods for graph neural networks. arXiv preprint arXiv:2206.09677, 2022.

[7] Dario Aragona, Luca Podo, Bardh Prenkaj, and Paola Velardi. Coronna: a deep sequential framework to predict epidemic spread. In Proceedings of the 36th Annual ACM Symposium on Applied Computing, pages 10–17, 2021.

[8] Alejandro Barredo Arrieta, Natalia Díaz-Rodríguez, Javier Del Ser, Adrien Bennetot, Siham Tabik, Alberto Barbado, Salvador García, Sergio Gil-López, Daniel Molina, Richard Benjamins, et al. Explainable artificial intelligence (xai): Concepts, taxonomies, opportunities and challenges toward responsible ai. Information fusion, 58:82–115, 2020.

[9] André Artelt and Barbara Hammer. On the computation of counterfactual explanations–a survey. arXiv preprint arXiv:1911.07749, 2019.

[10] Davide Bacciu, Federico Errica, and Alessio Micheli. Contextual graph markov model: A deep and generative approach to graph processing. In International Conference on Machine Learning, pages 294–303. PMLR, 2018.

[11] Mohit Bajaj, Lingyang Chu, Zi Yu Xue, Jian Pei, Lanjun Wang, Peter Cho-Ho Lam, and Yong Zhang. Robust counterfactual explanations on graph neural networks. Advances in Neural Information Processing Systems, 34, 2021.

[12] Dávid Bajusz, Anita Rácz, and Károly Héberger. Why is tanimoto index an appropriate choice for fingerprint-based similarity calculations? Journal of cheminformatics, 7(1):1–13, 2015.

[13] Monya Baker. 1,500 scientists lift the lid on reproducibility. Nature, 533(7604), 2016.

[14] Solon Barocas, Moritz Hardt, and Arvind Narayanan. Fairness in machine learning. Nips tutorial, 1:2, 2017.

[15] Ram B Basnet, Clayton Johnson, and Tenzin Doleck. Dropout prediction in moocs using deep learning and machine learning. Education and Information Technologies, pages 1–15, 2022.

[16] Michele Bevilacqua, Rexhina Blloshmi, and Roberto Navigli. One spring to rule them both: Symmetric amr semantic parsing and generation without a complex pipeline. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 35, pages 12564–12573, 2021.

[17] Alan Blair and Abdallah Saffidine. Ai surpasses humans at six-player poker. Science, 365(6456):864–865, 2019.

[18] Andrea Brennen. What do people really want when they say they want" explainable ai?" we asked 60 stakeholders. In Extended Abstracts of the 2020 CHI Conference on Human Factors in Computing Systems, pages 1–7, 2020.

[19] Jesse A Brown, Jeffrey D Rudie, Anita Bandrowski, John D Van Horn, and Susan Y Bookheimer. The ucla multimodal connectivity database: a web-based platform for brain connectivity matrix sharing and analysis. Frontiers in neuroinformatics, 6:28, 2012.

[20] Joan Bruna and X Li. Community detection with graph neural networks. stat, 1050:27, 2017.

[21] Ruth MJ Byrne. Counterfactuals in explainable artificial intelligence (xai): Evidence from human reasoning. In IJCAI, pages 6276–6282, 2019.

[22] Zhengdao Chen, Joan Bruna, and Lisha Li. Supervised community detection with line graph neural networks. In 7th International Conference on Learning Representations, ICLR 2019, 2019.

[23] Anoop Cherian, Suvrit Sra, Stephen Gould, and Richard Hartley. Non-linear temporal subspace representations for activity recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 2197–2206, 2018.

[24] Alejandra Bringas Colmenarejo, Luca Nannini, Alisa Rieger, Kristen M Scott, Xuan Zhao, Gourab K Patro, Gjergji Kasneci, and Katharina Kinder-Kurlanda. Fairness in agreement with european values: An interdisciplinary perspective on ai regulation. arXiv preprint arXiv:2207.01510, 2022.

[25] European Commission. Artificial intelligence for europe, 2018.

[26] European Commission. On artificial intelligence—a european approach to excellence and trust, 2020.

[27] European Commission. Regulatory framework proposal on artificial intelligence, 2021.

[28] Cameron Craddock, Yassine Benhajali, Carlton Chu, Francois Chouinard, Alan Evans, András Jakab, Budhachandra Singh Khundrakpam, John David Lewis, Qingyang Li, Michael Milham, et al. The neuro bureau preprocessing initiative: open sharing of preprocessed neuroimaging data and derivatives. Frontiers in Neuroinformatics, 7, 2013.

[29] Zhicheng Cui, Wenlin Chen, Yujie He, and Yixin Chen. Optimal action extraction for random forests and boosted trees. In Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining, pages 179–188, 2015.

[30] HIV Dataset. Dtp nci bulk data for download - nci dtp data - nci wiki, 2019.

[31] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. Advances in neural information processing systems, 29, 2016.

[32] Boris Delibašić, Milan Vukićević, Miloš Jovanović, and Milija Suknović. White-box or black-box decision tree algorithms: which to use in education? IEEE Transactions on Education, 56(3):287–291, 2012.

[33] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In NAACL-HLT (1), 2019.

[34] Venkatesulu Dondeti, Jyostna Devi Bodapati, Shaik Nagur Shareef, and Naralasetti Veeranjaneyulu. Deep convolution features in non-linear embedding space for fundus image classification. Rev. d'Intelligence Artif., 34(3):307–313, 2020.

[35] Elahi. Cheminformatics - dataset for molecular machine learning for drug discovery, 2019.

[36] Lukas Faber, Amin K Moghaddam, and Roger Wattenhofer. Contrastive graph neural network explanation. In Proceedings of the 37th Graph Representation Learning and Beyond Workshop at ICML 2020, page 28. International Conference on Machine Learning, 2020.

[37] Tom Fawcett. An introduction to roc analysis. Pattern recognition letters, 27(8):861–874, 2006.

[38] Emile Fiesler. Neural network classification and formalization. Computer Standards & Interfaces, 16(3):231–239, 1994.

[39] Alex A Freitas. Comprehensible classification models: a position paper. ACM SIGKDD explorations newsletter, 15(1):1–10, 2014.

[40] C Lee Giles, Kurt D Bollacker, and Steve Lawrence. Citeseer: An automatic citation indexing system. In Proceedings of the third ACM conference on Digital libraries, pages 89–98, 1998.

[41] Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. Neural message passing for quantum chemistry. In International conference on machine learning, pages 1263–1272. PMLR, 2017.

[42] Gustavo Grander, Luciano Ferreira da Silva, and Ernesto Del Rosário Santibanez Gonzalez. Big data as a value generator in decision support systems: A literature review. Revista de Gestão, 2021.

[43] Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks. In Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining, pages 855–864, 2016.

[44] Riccardo Guidotti. Counterfactual explanations and how to find them: literature review and benchmarking. Data Mining and Knowledge Discovery, pages 1–55, 2022.

[45] Riccardo Guidotti, Anna Monreale, Fosca Giannotti, Dino Pedreschi, Salvatore Ruggieri, and Franco Turini. Factual and counterfactual explanations for black box decision making. IEEE Intelligent Systems, 34(6):14–23, 2019.

[46] Riccardo Guidotti, Anna Monreale, Salvatore Ruggieri, Franco Turini, Fosca Giannotti, and Dino Pedreschi. A survey of methods for explaining black box models. ACM computing surveys (CSUR), 51(5):1–42, 2018.

[47] David Gunning. Explainable artificial intelligence (xai). Defense advanced research projects agency (DARPA), nd Web, 2(2):1, 2017.

[48] Yu He, Yangqiu Song, Jianxin Li, Cheng Ji, Jian Peng, and Hao Peng. Hetespaceywalk: A heterogeneous spacey random walk for heterogeneous information network embedding. In Proceedings of the 28th ACM International Conference on Information and Knowledge Management, pages 639–648, 2019.

[49] Peter Henderson, Riashat Islam, Philip Bachman, Joelle Pineau, Doina Precup, and David Meger. Deep reinforcement learning that matters. In Proceedings of the AAAI conference on artificial intelligence, volume 32, 2018.

[50] Fenyu Hu, Yanqiao Zhu, Shu Wu, Liang Wang, and Tieniu Tan. Hierarchical graph convolutional networks for semi-supervised node classification. In IJCAI, 2019.

[51] Barry D Hughes. Random walks and random environments. Bulletin of the American Mathematical Society, 35(4):347–349, 1998.

[52] Sarthak Jain and Byron C Wallace. Attention is not explanation. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pages 3543–3556, 2019.

[53] Shuiwang Ji, Wei Xu, Ming Yang, and Kai Yu. 3d convolutional neural networks for human action recognition. IEEE transactions on pattern analysis and machine intelligence, 35(1):221–231, 2012.

[54] Weiwei Jiang and Jiayun Luo. Graph neural network for traffic forecasting: A survey. Expert Systems with Applications, page 117921, 2022.

[55] Xiaodong Jiang, Ronghang Zhu, Sheng Li, and Pengsheng Ji. Co-embedding of nodes and edges with graph neural networks. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2020.

[56] Alexandros Karatzoglou and Balázs Hidasi. Deep learning for recommender systems. In Proceedings of the eleventh ACM conference on recommender systems, pages 396–397, 2017.

[57] Jeroen Kazius, Ross McGuire, and Roberta Bursi. Derivation and validation of toxicophores for mutagenicity prediction. Journal of medicinal chemistry, 48(1):312–320, 2005.

[58] Kristian Kersting, Nils M Kriege, Christopher Morris, Petra Mutzel, and Marion Neumann. Benchmark data sets for graph kernels. URL http://graphkernels. cs. tu-dortmund. de, 2016.

[59] Jongmin Kim, Taesup Kim, Sungwoong Kim, and Chang D Yoo. Edge-labeling graph neural network for few-shot learning. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pages 11–20, 2019.

[60] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In 5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings. OpenReview.net, 2017.

[61] Johannes Klicpera, Janek Groß, and Stephan Günnemann. Directional message passing for molecular graphs. arXiv preprint arXiv:2003.03123, 2020.

[62] Ismail Kola and John Landis. Can the pharmaceutical industry reduce attrition rates? Nature reviews Drug discovery, 3(8):711–716, 2004.

[63] Maxim Kovalev, Lev Utkin, Frank Coolen, and Andrei Konstantinov. Counterfactual explanation of machine learning survival models. Informatica, 32(4):817–847, 2021.

[64] Mario Krenn, Florian Häse, AkshatKumar Nigam, Pascal Friederich, and Alan Aspuru-Guzik. Self-referencing embedded strings (selfies): A 100% robust molecular string representation. Machine Learning: Science and Technology, 1(4):045024, 2020.

[65] Shishir Kulkarni, Jay Ketan Katariya, and Katerina Potika. Glovenor: Glove for node representations with second order random walks. In 2020 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM), pages 536–543. IEEE, 2020.

[66] Orestis Lampridis, Riccardo Guidotti, and Salvatore Ruggieri. Explaining sentiment classification with synthetic exemplars and counter-exemplars. In International Conference on Discovery Science, pages 357–373. Springer, 2020.

[67] Jack Lanchantin, Arshdeep Sekhon, and Yanjun Qi. Neural message passing for multi-label classification. In Joint European Conference on Machine Learning and Knowledge Discovery in Databases, pages 138–163. Springer, 2019.

[68] Tommaso Lanciano, Francesco Bonchi, and Aristides Gionis. Explainable classification of brain networks via contrast subgraphs. In Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pages 3308–3318, 2020.

[69] John Boaz Lee, Ryan Rossi, and Xiangnan Kong. Graph classification using structural attention. In Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pages 1666–1674, 2018.

[70] Yang Li and Yi Pan. A novel ensemble deep learning model for stock prediction based on stock prices and news. International Journal of Data Science and Analytics, 13(2):139–149, 2022.

[71] Yujia Li, Daniel Tarlow, Marc Brockschmidt, and Richard S. Zemel. Gated graph sequence neural networks. In Yoshua Bengio and Yann LeCun, editors, 4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings, 2016.

[72] Wanyu Lin, Hao Lan, and Baochun Li. Generative causal explanations for graph neural networks. In International Conference on Machine Learning, pages 6666–6679. PMLR, 2021.

[73] Zachary C Lipton. The mythos of model interpretability: In machine learning, the concept of interpretability is both important and slippery. Queue, 16(3):31–57, 2018.

[74] Octavio Loyola-González. Black-box vs. white-box: Understanding their advantages and weaknesses from a practical point of view. IEEE Access, 7:154096–154113, 2019.

[75] Ana Lucic, Maartje A Ter Hoeve, Gabriele Tolomei, Maarten De Rijke, and Fabrizio Silvestri. Cf-gnnexplainer: Counterfactual explanations for graph neural networks. In International Conference on Artificial Intelligence and Statistics, pages 4499–4511. PMLR, 2022.

[76] Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. Advances in neural information processing systems, 30, 2017.

[77] Dongsheng Luo, Wei Cheng, Dongkuan Xu, Wenchao Yu, Bo Zong, Haifeng Chen, and Xiang Zhang. Parameterized explainer for graph neural network. Advances in neural information processing systems, 33:19620–19631, 2020.

[78] Ruixin Ma, Fangqing Guo, Zeyang Li, and Liang Zhao. Knowledge graph random neural networks for recommender systems. Expert Systems with Applications, 201:117120, 2022.

[79] Xiaoxiao Ma, Jia Wu, Shan Xue, Jian Yang, Chuan Zhou, Quan Z Sheng, Hui Xiong, and Leman Akoglu. A comprehensive survey on graph anomaly detection with deep learning. IEEE Transactions on Knowledge and Data Engineering, 2021.

[80] Lorenzo Madeddu and Giovanni Stilo. Deep learning methods for network biology. In Deep Learning In Biology And Medicine, pages 197–246. World Scientific, 2022.

[81] Lorenzo Madeddu, Giovanni Stilo, and Paola Velardi. A feature-learning-based method for the disease-gene prediction problem. International Journal of Data Mining and Bioinformatics, 24(1):16–37, 2020.

[82] Marta Marchiori Manerba and Riccardo Guidotti. Investigating debiasing effects on classification and explainability. In Proceedings of the 2022 AAAI/ACM Conference on AI, Ethics, and Society, AIES '22, page 468–478, New York, NY, USA, 2022. Association for Computing Machinery.

[83] Ines Filipa Martins, Ana L Teixeira, Luis Pinheiro, and Andre O Falcao. A bayesian approach to in silico blood-brain barrier penetration modeling. Journal of chemical information and modeling, 52(6):1686–1697, 2012.

[84] Ninareh Mehrabi, Fred Morstatter, Nripsuta Saxena, Kristina Lerman, and Aram Galstyan. A survey on bias and fairness in machine learning. ACM Computing Surveys (CSUR), 54(6):1–35, 2021.

[85] Tim Miller. Explanation in artificial intelligence: Insights from the social sciences. Artificial intelligence, 267:1–38, 2019.

[86] Christoph Molnar, Giuseppe Casalicchio, and Bernd Bischl. Interpretable machine learning–a brief history, state-of-the-art and challenges. In Joint European Conference on Machine Learning and Knowledge Discovery in Databases, pages 417–431. Springer, 2020.

[87] Federico Monti, Davide Boscaini, Jonathan Masci, Emanuele Rodola, Jan Svoboda, and Michael M Bronstein. Geometric deep learning on graphs and manifolds using mixture model cnns. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 5115–5124, 2017.

[88] James Mullenbach, Sarah Wiegreffe, Jon Duke, Jimeng Sun, and Jacob Eisenstein. Explainable prediction of medical codes from clinical text. In Proceedings of NAACL-HLT, pages 1101–1111, 2018.

[89] & Medicine National Academies of Sciences, Engineering. Reproducibility and replicability in science. 2019.

[90] AkshatKumar Nigam, Robert Pollice, Mario Krenn, Gabriel dos Passos Gomes, and Alan Aspuru-Guzik. Beyond generative models: superfast traversal, optimization, novelty, exploration and discovery (stoned) algorithm for molecules using selfies. Chemical science, 2021.

[91] Kathryn M Nowotny, Zinzi Bailey, and Lauren Brinkley-Rubinstein. The contribution of prisons and jails to us racial disparities during covid-19, 2021.

[92] Danilo Numeroso and Davide Bacciu. Explaining deep graph networks with molecular counterfactuals. arXiv preprint arXiv:2011.05134, 2020.

[93] Danilo Numeroso and Davide Bacciu. Meg: Generating molecular counterfactual explanations for deep graph networks. In 2021 International Joint Conference on Neural Networks (IJCNN), pages 1–8. IEEE, 2021.

[94] Ziad Obermeyer, Brian Powers, Christine Vogeli, and Sendhil Mullainathan. Dissecting racial bias in an algorithm used to manage the health of populations. Science, 366(6464):447–453, 2019.

[95] World Health Organization et al. Ethics and governance of artificial intelligence for health: Who guidance. 2021.

[96] Guansong Pang, Chunhua Shen, Longbing Cao, and Anton Van Den Hengel. Deep learning for anomaly detection: A review. ACM Computing Surveys (CSUR), 54(2):1–38, 2021.

[97] Aldo Pareja, Giacomo Domeniconi, Jie Chen, Tengfei Ma, Toyotaro Suzumura, Hiroki Kanezashi, Tim Kaler, Tao Schardl, and Charles Leiserson. Evolvegcn: Evolving graph convolutional networks for dynamic graphs. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 34, pages 5363–5370, 2020.

[98] European Parliament. Regulation (eu) 2016/679 of the european parliament and of the council of 27 april 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing directive 95/46/ec (general data protection regulation) (text with eea relevance). `https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX:32016R0679`, Apr 2016. [Online; accessed 11-January-2022].

[99] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. Deepwalk: Online learning of social representations. In Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining, pages 701–710, 2014.

[100] Dana Pessach and Erez Shmueli. A review on fairness in machine learning. ACM Computing Surveys (CSUR), 55(3):1–44, 2022.

[101] Jeremy Petch, Shuang Di, and Walter Nelson. Opening the black box: the promise and limitations of explainable machine learning in cardiology. Canadian Journal of Cardiology, 2021.

[102] Joelle Pineau, Philippe Vincent-Lamarre, Koustuv Sinha, Vincent Larivière, Alina Beygelzimer, Florence d'Alché Buc, Emily Fox, and Hugo Larochelle. Improving reproducibility in machine learning research: a report from the neurips 2019 reproducibility program. Journal of Machine Learning Research, 22, 2021.

[103] Mario Alfonso Prado-Romero and Giovanni Stilo. Gretel: Graph counterfactual explanation evaluation framework. In Proceedings of the 31st ACM International Conference on Information and Knowledge Management, CIKM '22, New York, NY, USA, 2022. Association for Computing Machinery.

[104] Bardh Prenkaj, Damiano Distante, Stefano Faralli, and Paola Velardi. Hidden space deep sequential risk prediction on student trajectories. Future Generation Computer Systems, 125:532–543, 2021.

[105] Bardh Prenkaj, Paola Velardi, Damiano Distante, and Stefano Faralli. A reproducibility study of deep and surface machine learning methods for human-related trajectory prediction. In Proceedings of the 29th ACM International Conference on Information & Knowledge Management, pages 2169–2172, 2020.

[106] Jiezhong Qiu, Jian Tang, Hao Ma, Yuxiao Dong, Kuansan Wang, and Jie Tang. Deepinf: Social influence prediction with deep learning. In Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining, pages 2110–2119, 2018.

[107] Gabrielle Ras, Ning Xie, Marcel van Gerven, and Derek Doran. Explainable deep learning: A field guide for the uninitiated. Journal of Artificial Intelligence Research, 73:329–397, 2022.

[108] David Rogers and Mathew Hahn. Extended-connectivity fingerprints. Journal of chemical information and modeling, 50(5):742–754, 2010.

[109] Yu Rong, Wenbing Huang, Tingyang Xu, and Junzhou Huang. Dropedge: Towards deep graph convolutional networks on node classification. In International Conference on Learning Representations, 2019.

[110] Krzysztof Sadowski, Michał Szarmach, and Eddie Mattia. Dimensionality reduction meets message passing for graph node embeddings. arXiv preprint arXiv:2202.00408, 2022.

[111] Alberto Sanfeliu and King-Sun Fu. A distance measure between attributed relational graphs for pattern recognition. IEEE transactions on systems, man, and cybernetics, (3):353–362, 1983.

[112] Hamed Sarvari, Carlotta Domeniconi, Bardh Prenkaj, and Giovanni Stilo. Unsupervised boosting-based autoencoder ensembles for outlier detection. In Pacific-Asia Conference on Knowledge Discovery and Data Mining, pages 91–103. Springer, 2021.

[113] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. The graph neural network model. IEEE transactions on neural networks, 20(1):61–80, 2008.

[114] Julian Schrittwieser, Ioannis Antonoglou, Thomas Hubert, Karen Simonyan, Laurent Sifre, Simon Schmitt, Arthur Guez, Edward Lockhart, Demis Hassabis, Thore Graepel, et al. Mastering atari, go, chess and shogi by planning with a learned model. Nature, 588(7839):604–609, 2020.

[115] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. nature, 529(7587):484–489, 2016.

[116] David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. Mastering the game of go without human knowledge. nature, 550(7676):354–359, 2017.

[117] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556, 2014.

[118] Theodoros Sofianos, Alessio Sampieri, Luca Franco, and Fabio Galasso. Space-time-separable graph convolutional network for pose forecasting. In Proceedings of the IEEE/CVF International Conference on Computer Vision, pages 11209–11218, 2021.

[119] Suraj Srinivas, Akshayvarun Subramanya, and R Venkatesh Babu. Training sparse neural networks. In 2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), pages 455–462. IEEE, 2017.

[120] U Steinhoff, A Wiesmaier, and R Araújo. The state of the art in dns spoofing. In Proc. 4th Intl. Conf. Applied Cryptography and Network Security (ACNS), 2006.

[121] Ilia Stepin, Jose M Alonso, Alejandro Catala, and Martín Pereira-Fariña. A survey of contrastive and counterfactual explanation generation methods for explainable artificial intelligence. IEEE Access, 9:11974–12001, 2021.

[122] Jianyong Sun, Wei Zheng, Qingfu Zhang, and Zongben Xu. Graph neural network encoding for community detection in attribute networks. IEEE Transactions on Cybernetics, 2021.

[123] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander A Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In Thirty-first AAAI conference on artificial intelligence, 2017.

[124] Juntao Tan, Shijie Geng, Zuohui Fu, Yingqiang Ge, Shuyuan Xu, Yunqi Li, and Yongfeng Zhang. Learning and evaluating graph neural network explanations based on counterfactual and factual reasoning. In Proceedings of the ACM Web Conference 2022, pages 1018–1027, 2022.

[125] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. Line: Large-scale information network embedding. In Proceedings of the 24th international conference on world wide web, pages 1067–1077, 2015.

[126] Rachael Tatman, Jake VanderPlas, and Sohier Dane. A practical taxonomy of reproducibility for machine learning research. 2018.

[127] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In Advances in neural information processing systems, pages 5998–6008, 2017.

[128] Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In 6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings. OpenReview.net, 2018.

[129] Ilya Verenich, Marlon Dumas, Marcello La Rosa, and Hoang Nguyen. Predicting process performance: A white-box approach based on process models. Journal of Software: Evolution and Process, 31(6):e2170, 2019.

[130] Ilya Verenich, Hoang Nguyen, Marcello La Rosa, and Marlon Dumas. White-box prediction of process performance indicators via flow analysis. In Proceedings of the 2017 International Conference on Software and System Process, pages 85–94, 2017.

[131] Hanuman Verma, Saurav Mandal, and Akshansh Gupta. Temporal deep learning architecture for prediction of covid-19 cases in india. Expert Systems with Applications, 195:116611, 2022.

[132] Nitika Verma, Edmond Boyer, and Jakob Verbeek. Feastnet: Feature-steered graph convolutions for 3d shape analysis. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 2598–2606, 2018.

[133] Sahil Verma, John Dickerson, and Keegan Hines. Counterfactual explanations for machine learning: A review. arXiv preprint arXiv:2010.10596, 2020.

[134] Ruben Vicente-Saez and Clara Martinez-Fuentes. Open science now: A systematic literature review for an integrated definition. Journal of business research, 88:428–436, 2018.

[135] Carlos J Villagrá-Arnedo, Francisco J Gallego-Durán, Faraón Llorens-Largo, Patricia Compañ-Rosique, Rosana Satorre-Cuerda, and Rafael Molina-Carmona. Improving the expressiveness of black-box models for predicting student performance. Computers in Human Behavior, 72:621–631, 2017.

[136] Minh Vu and My T Thai. Pgm-explainer: Probabilistic graphical model explanations for graph neural networks. Advances in neural information processing systems, 33:12225–12235, 2020.

[137] Sandra Wachter, Brent Mittelstadt, and Chris Russell. Counterfactual explanations without opening the black box: Automated decisions and the gdpr. Harv. JL & Tech., 31:841, 2017.

[138] Nikil Wale, Ian A Watson, and George Karypis. Comparison of descriptor spaces for chemical compound retrieval and classification. Knowledge and Information Systems, 14(3):347–375, 2008.

[139] Binghui Wang, Jinyuan Jia, and Neil Zhenqiang Gong. Semi-supervised node classification on graphs: Markov random fields vs. graph neural networks. In AAAI, pages 10093–10101, 2021.

[140] Junkang Wei, Siyuan Chen, Licheng Zong, Xin Gao, and Yu Li. Protein–rna interaction prediction with deep learning: structure matters. Briefings in bioinformatics, 23(1):bbab540, 2022.

[141] Geemi P Wellawatte, Aditi Seshadri, and Andrew D White. Model agnostic generation of counterfactual explanations for molecules. Chemical science, 13(13):3697–3705, 2022.

[142] Max Welling and Thomas N Kipf. Semi-supervised classification with graph convolutional networks. In J. International Conference on Learning Representations (ICLR 2017), 2016.

[143] David R Williams and Toni D Rucker. Understanding and addressing racial disparities in health care. Health care financing review, 21(4):75, 2000.

[144] Olivier J Wouters, Martin McKee, and Jeroen Luyten. Estimated research and development investment needed to bring a new medicine to market, 2009-2018. Jama, 323(9):844–853, 2020.

[145] Haoran Wu, Wei Chen, Shuang Xu, and Bo Xu. Counterfactual supporting facts extraction for explainable medical record based diagnosis with graph network. In Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 1942–1955, 2021.

[146] Jun Wu, Jingrui He, and Jiejun Xu. Net: Degree-specific graph neural networks for node and graph classification. In Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pages 406–415, 2019.

[147] Zhenqin Wu, Bharath Ramsundar, Evan N Feinberg, Joseph Gomes, Caleb Geniesse, Aneesh S Pappu, Karl Leswing, and Vijay Pande. Moleculenet: a benchmark for molecular machine learning. Chemical science, 9(2):513–530, 2018.

[148] Mingying Xu, Junping Du, Zhe Xue, Zeli Guan, Feifei Kou, and Lei Shi. A scientific research topic trend prediction model based on multi-lstm and graph convolutional network. International Journal of Intelligent Systems, 2022.

[149] Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L Hamilton, and Jure Leskovec. Graph convolutional neural networks for web-scale recommender systems. In Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining, pages 974–983, 2018.

[150] Zhitao Ying, Dylan Bourgeois, Jiaxuan You, Marinka Zitnik, and Jure Leskovec. Gnnexplainer: Generating explanations for graph neural networks. Advances in neural information processing systems, 32, 2019.

[151] Jiaxuan You, Jonathan M Gomes-Selman, Rex Ying, and Jure Leskovec. Identity-aware graph neural networks. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 35, pages 10737–10745, 2021.

[152] Hao Yuan, Jiliang Tang, Xia Hu, and Shuiwang Ji. Xgnn: Towards model-level explanations of graph neural networks. In Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pages 430–438, 2020.

[153] Hao Yuan, Haiyang Yu, Shurui Gui, and Shuiwang Ji. Explainability in graph neural networks: A taxonomic survey. arXiv preprint arXiv:2012.15445, 2020.

[154] Hao Yuan, Haiyang Yu, Jie Wang, Kang Li, and Shuiwang Ji. On explainability of graph neural networks via subgraph explorations. In International Conference on Machine Learning, pages 12241–12252. PMLR, 2021.

[155] Shunjie Yuan, Chao Wang, Qi Jiang, and Jianfeng Ma. Community detection with graph neural network using markov stability. In 2022 International Conference on Artificial Intelligence in Information and Communication (ICAIIC), pages 437–442. IEEE, 2022.

[156] Wayne W Zachary. An information flow model for conflict and fission in small groups. Journal of anthropological research, 33(4):452–473, 1977.

[157] Muhan Zhang and Yixin Chen. Link prediction based on graph neural networks. Advances in neural information processing systems, 31, 2018.

[158] Si Zhang, Hanghang Tong, Jiejun Xu, and Ross Maciejewski. Graph convolutional networks: a comprehensive review. Computational Social Networks, 6(1):1–23, 2019.

[159] Qichang Zhao, Haochen Zhao, Kai Zheng, and Jianxin Wang. Hyperattentiondti: improving drug–protein interaction prediction by sequence-based deep learning with attention mechanism. Bioinformatics, 38(3):655–662, 2022.

[160] Tianxiang Zhao, Xiang Zhang, and Suhang 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, pages 833–841, 2021.

[161] Zhiqiang Zhong, Cheng-Te Li, and Jun Pang. Hierarchical message-passing graph neural networks. arXiv preprint arXiv:2009.03717, 2020.

[162] Fan Zhou, Chengtai Cao, Kunpeng Zhang, Goce Trajcevski, Ting Zhong, and Ji Geng. Meta-gnn: On few-shot node classification in graph meta-learning. In Proceedings of the 28th ACM International Conference on Information and Knowledge Management, pages 2357–2360, 2019.

[163] Zhenpeng Zhou, Steven Kearnes, Li Li, Richard N Zare, and Patrick Riley. Optimization of molecules via deep reinforcement learning. Scientific Reports, 9(1):1–10, 2019.

[164] Barret Zoph, Vijay Vasudevan, Jonathon Shlens, and Quoc V Le. Learning transferable architectures for scalable image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 8697–8710, 2018.

[165] Daniel Zügner, Amir Akbarnejad, and Stephan Günnemann. Adversarial attacks on neural networks for graph data. In Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining, pages 2847–2856, 2018.