Enhancing Causal Effect Estimation from Networked Observational Data by Denoising Graph

Changlong Shi[1], Qiang Huang[4], Huiyan Sun [1], Yi Chang [123]
[1]School of Artificial Intelligence, Jilin University,
[2]International Center of Future Science, Jilin University
[3]Engineering Research Center of Knowledge-Driven Human-Machine Intelligence, MOE, China
[4]Mohamed bin Zayed University of Artificial Intelligence
shicl22@mails.jlu.edu.cn, qiang.huang@mbzuai.ac.ae,{yichang,huiyansun}@jlu.edu.cn,

Abstract

Estimating causal effects from observational data has garnered significant attention in recent years, as it facilitates decision-making in various domains, such as healthcare, economy, and social science. Recently, many studies have focused on networked observational data, utilizing the auxiliary network structure to infer hidden confounders for improved performance in causal effect estimation. However, networked observational data often contains noise in practical scenarios and existing methods often experience a severe performance decline when the edge noise is present in terms of the graph structure, leading to disrupted and biased causal estimation. Thus, denoising the collected graph and getting the optimal network structure is critical for precise causal effect estimation. In this paper, we propose a novel approach, referred as EDge reweIghTing Of multi-subgRaph (EDITOR), to eliminate the graph noise for robust causal effect estimation. Specifically, by utilizing graph neural network, EDITOR partitions the perturbed graph into distinct subgraphs based on the edge type and learning their importance weights for each subgraph. By doing so, our method effectively uncovers the clean graph structure from perturbed networked data while preserving the underlying causal information. Extensive experiments are conducted over different datasets and perturbations, demonstrating that the proposed methods achieve significantly higher performance and robustness than state-of-the-art causal effect estimation methods.

Introduction

The study of causal effect estimation is significantly important for the development of diverse fields such as economics (Hünermund and Bareinboim 2023), recommender systems (Gao et al. 2024), and health care (Prosperi et al. 2020). It aims to identify confounding variables and discover causal relationships between treatment and outcome variables. With the continuous advancement of machine learning, an increasing number of methods (Huang et al. 2022a; Cheng et al. 2020; Johansson, Shalit, and Sontag 2016; Guo et al. 2020) have emerged that they utilize observational data for causal effect estimation as a substitute for randomized controlled trials (RCTs) (Gui et al. 2015; Yuan, Altenburger, and Kooti 2021). Observational data typically comprises numerous entities and exhibits rich features. In certain situations, entities are naturally interconnected, and these connections can be represented as a network structure. For instance, users in social networks are connected through friendships, and in article networks, connections are established between authors through citation relationships. Networks frequently encapsulate valuable causal information and help improve the precision of effect estimation. For example, hidden variables, such as social status and interpersonal relationships, which may not be directly observable from the features, can often be inferred through these relationship networks.

Through incorporating relational networks along with instance features, many works have been shown to be powerful in causal effect estimation (Ma and Li 2022; Ma and Tresp 2021). However, these methods' performance can be significantly affected by the presence of edge noise in the network, as such disturbances can readily undermine the inherent causal information present within the network. For instance, in the process of decision making related to the selection of costly medical treatments, perturbations may occur that individuals with lower social status can be erroneously associated with those of higher social status in the relationship network. As a result, the model may inaccurately estimate their social status, significantly impairing the model's ability to make precise judgments about their treatment choices. In summary, the networked observational data frequently exhibits inherent noise in practical scenarios, and denoising the graph structure is crucial for causal analysis. Existing graph structure denoising methods (Jin et al. 2020; Dai et al. 2022) are not customized for causal effect estimation tasks, resulting in suboptimal performance. Therefore, it is important to develop a method that can effectively eliminate noise from networked observational data while simultaneously recovering the underlying causal information for robust causal effect estimation.

Previous works (McPherson, Smith-Lovin, and Cook 2001; Vaughan, Kipp, and Gao 2007) have demonstrated that nodes in a network that exhibit similar features and behaviors are more likely to be connected. Consequently, we propose the assumption that users in social networks who display similar behaviors (choose the same treatment) are more likely to be connected. This results in a higher intra-group edge density (treatment or control group) compared to the inter-group edge density in the network structure. In

Copyright © 2025, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

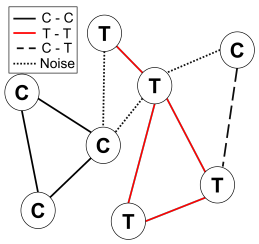


Figure 1: Overview of Network Edge Structure. Where T/C represents instances with treatment = 1/0.

contrast, perturbed edges are typically random and do not adhere to these properties (Wu et al. 2019). An intuitive example is shown in Figure 1. It can be observed that in a social network without noise, the intra-group edge density is higher than the inter-group edge density, whereas noise edges are random. We aim to utilize this property to distinguish and remove noise edges. To achieve this, We propose a novel approach called EDge reweIghTing Of multi-subgRaph (EDITOR), which effectively learns the clean graph structure from perturbed networks while preserving the underlying causal information. Our method categorizes the edges into three types: intra-treatment-group edges, intra-control-group edges, and inter-group edges, corresponding to T - T, C - C, and T - C in Figure 1, respectively. EDITOR partitions the graph into distinct subgraphs based on edge types and subsequently learns their respective importance weights. It identifies noise edges and denoises the graph by assigning smaller weights to those edges that violate our assumptions. To further refine our model, we impose constraints on the learned graph structure, such as symmetry preservation and feature smoothness. These constraints ensure that the resulting structure retains important graph properties.

The primary contributions of this work can be summarized as follows:

- We formulate a novel research problem, recovering causal relationships in networked data when subjected to the perturbations. To our knowledge, this is the first work specifically focused on graph data denoising for causal effect estimation.
- We propose the method EDITOR under the assumption that different types of edges within a network often exhibit distinct characteristics. This method uncovers the clean graph structure from perturbed network data and restores the causal information within the network by learning edge weight matrices for distinct subgraphs.
- We conducted comprehensive experiments to demonstrate that the proposed EDITOR outperforms state-of-the-art methods in estimating causal effects, even in scenarios where the network structure incorporates noise or undergoes perturbations.

Related Work

In this section, we provide an overview of the relevant works on causal inference and graph structure denoising.

Causal Inference

In recent years, due to the rapid advancements in machine learning technology, there has been significant attention directed toward estimating causal effects using observational data (Zeng and Wang 2022; Chu and Li 2023; Veitch, Wang, and Blei 2019). CFR (Shalit, Johansson, and Sontag 2017) is based on the strong ignorability assumption, it maps the original data features to the corresponding representation space to obtain the representation of confounders. During training, it minimizes the prediction error of the outcome and the IPM which measures the distributional imbalance between the control and treatment groups. TARNet (Shalit, Johansson, and Sontag 2017) is a variant of CFR without the IPM. CEVAE (Louizos et al. 2017) utilizes the variational autoencoder to estimate causal effect. DeR-CFR (Wu et al. 2023) decomposes instrumental, confounding, and adjustment factors by managing correlations between variables, while simultaneously learning counterfactual regression to estimate treatment effects in observational studies. DESCN (Zhong et al. 2022) leveraged multi-task learning within a crossover network to obtain comprehensive insights into treatment propensity, response, and hidden treatment effects. ESCFR (Wang et al. 2024) utilizes optimal transport to balance the covariate representations across different treatment groups. Many other works achieve great performance in causal effect estimation by incorporating relational networks along with instance features. ND (Guo, Li, and Liu 2020) initially learns hidden confounders by leveraging the relationships between entities in observational networked data, uses graph convolutional network (GCN) to learn the representation of confounders, and employs IPM for representation distribution balancing. GIAL (Chu, Rathbun, and Li 2021) addresses the graph structure imbalance issue between control and treatment groups in networked data by employing Adversarial Learning. NetEst (Jiang and Sun 2022) utilizes adversarial learning to bridge the distribution gaps of the objective function between the commonly used GNN and the causal effect estimation. These methods have shown encouraging performance. However, their performance may significantly degrade when the graph structure is perturbed. Therefore, we hope to investigate an approach that can effectively estimate causal effects even when the network structure is perturbed.

Graph Structure Denoising

Graph learning has gained significant popularity in recent years due to its wide applicability to various real-world problems (Schlichtkrull et al. 2018; Veličković et al. 2018; Wu et al. 2020; Atwood and Towsley 2016; Huang et al. 2022b; Ying et al. 2018). Many data in the real world can naturally be modeled in the form of graphs. GNNs can directly operate on graphs and leverage their structural information, leading to considerable success in addressing machine learning problems based on graphs. However, many studies have shown that graph neural networks are vulnerable to noise in graph structures (Dai et al. 2018; Jin et al. 2021; Zügner, Akbarnejad, and Günnemann 2018). Hence, many methods focus on learning clean network structures by reducing attention to interfering edges. (Wu et al. 2019)

observed that attackers tend to connect nodes with different features and proposed removing links between dissimilar nodes to learn a clean graph structure. (Entezari et al. 2020) point out that network attacks cause changes in the high-order spectrum of graphs and suggest using low-order approximations for graph preprocessing. (Jin et al. 2020) propose the Pro-GNN, while learning GNN parameters, explores essential graph properties to recover a clean graph, enabling the proposed model to extract intrinsic structures from perturbed graphs under different attacks. RS-GNN (Dai et al. 2022) adopts the edges in the noisy graph as supervision to obtain a denoised and densified graph.

Different from the aforementioned approaches, our method is designed for causal effect estimation tasks. It considers the distinctions among various types of edges and respectively learns their importance weights to obtain the clean graph structure from the perturbed network.

Problem Statement

Before presenting the problem statement, we introduce some notations and basic concepts. The Frobenius norm of a matrix $\mathbf{A}$ is defined by $||\mathbf{A}||_F^2 = \Sigma_{ij}\mathbf{A}_{ij}^2$. We use $\mathcal{G}(\mathbf{A}, \mathbf{X})$ to denote an undirected graph, where $\mathbf{A} \in \mathbb{R}^{n\times n}$ is the adjacency matrix of the graph, if there has an edge between the i-th instance and the j-th instance, $\mathbf{A}_{ij} = \mathbf{A}_{ji} = 1$, otherwise $\mathbf{A}_{ij} = \mathbf{A}_{ji} = 0$. And $\mathbf{X} = [\mathbf{x}_1, ..., \mathbf{x}_n] \in \mathbb{R}^{n\times d}$ is the feature matrix of instances, where $\mathbf{x}_i$ is the feature vector of instance i. Let $\mathbf{T} = \{t_1, ..., t_n\}$ denote a set of individual treatment assignment, where n is the number of instances, t_i is the treatment assignment of the i-th instance, $t_i = 0$ ($t_i = 1$) means the i-th instance is under controlled (treated). The potential outcome $\mathbf{Y} = \{y_1^{t_1}, ..., y_n^{t_n}\}$ is the possible outcome with different treatment, $y_i^{t_i}$ denotes the outcome value of instance i when treatment is t_i. In networked observational data, only one of the two potential outcomes can be observed, the observed outcome is the factual outcome $y_i^f = y_i^{t_i}$, and the unobserved potential outcome is the counterfactual outcome $y_i^{cf} = y_i^{1-t_i}$. The main challenge of learning individual treatment effects is the inference of counterfactual outcomes. The Individual Treatment Effect (ITE) quantifies the change in the outcome of an instance due to the treatment. The ITE of the i-th instance and the Average Treatment Effect (ATE) are defined as:

$$ITE_i = y_i^1 - y_i^0, \tag{1a}$$

$$ATE = \frac{1}{n}\sum_{i=1}^{n}(ITE_i). \tag{1b}$$

With the aforementioned notations and concepts, we can state the problem that we aim to study in this work as follow: For a given graph $\mathcal{G}(\mathbf{A}, \mathbf{X})$, the graph structure $\mathbf{A}$ has been perturbed. We aim to learn a clean structure $\hat{\mathbf{A}}$ that contains complete causal information, and incorporate $\hat{\mathbf{A}}$ along with the feature matrix $\mathbf{X}$ to estimate the ITEs. Although estimating the exact ITEs is challenging and often infeasible, we follow the terminology used in previous work (Jiang and Sun 2022; Guo, Li, and Liu 2020). In practice, we focus on approaches approximating these effects, which can be interpreted as estimating the Conditional Average Treatment Effects (CATEs).

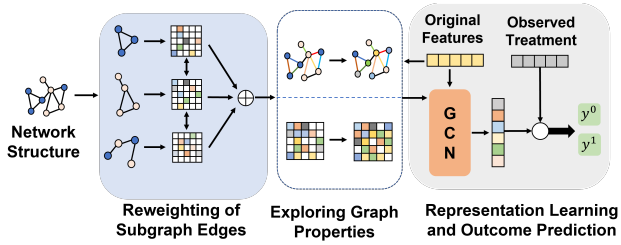


Figure 2: The overall depiction of framework EDITOR.

The Proposed Framework

The noise of graph structure can significantly impair the performance of existing methods for causal effects estimation. Thus, it is essential to design a new method that can eliminate the perturbed edges among the graph structure and preserve the underlying causal information. In this section, we propose the EDITOR aims to address this problem. The illustration of the framework is shown in Figure 2, our model primarily consists of three components: Reweighting of Subgraph Edges, Exploring Graph Properties, Representation Learning and Outcome Prediction. In the following subsections, we provide detailed explanations of the proposed framework.

Reweighting of Subgraph Edges

Reweighting of Subgraph Edges aims to denoise the graph by learning importance weights for distinct types of edges, assigning higher weights to important edges and lower weights to perturbed edges. For the causal effect estimation task, as shown in Figure 1, we partition the adjacency matrix $\mathbf{A}$ of graph $\mathcal{G}(\mathbf{A}, \mathbf{X})$ into three subgraphs based on the types of edges, denoted as $\mathbf{A}_c$, $\mathbf{A}_t$ and $\mathbf{A}_{ct}$. These subgraphs respectively correspond to the edges of node pairs within the control group, the edges of node pairs within the treatment group, and the edges of node pairs between the control and treatment group. Then we build three learnable matrices $\mathbf{W}_{ct}$, $\mathbf{W}_c$ and $\mathbf{W}_t$ to learn the importance weights of each edge type respectively. The weighted graph structure can be formulated as$\hat{\mathbf{A}}_* = \mathbf{A}_* \cdot \mathbf{W}_*$, where $* \in \{c, t, ct\}$ is the edge type. Following our assumption, closely connected instances often exhibit similar behaviors (McPherson, Smith-Lovin, and Cook 2001; Vaughan, Kipp, and Gao 2007; Kipf and Welling 2017), resulting in higher edge density within treatment (control) groups compared to between groups in the network structure, whereas noise edges are usually random. Therefore, we expect that the weight matrix can leverage this information to distinguish noise edges by assigning them smaller weights. We define the following edge density distribution control function to achieve this.

$$\mathcal{L}_d = max(0, d(\hat{\mathbf{A}}_{ct}) - \frac{d(\hat{\mathbf{A}}_c) + d(\hat{\mathbf{A}}_t)}{\tau}), \quad (2a)$$

$$d(\hat{\mathbf{A}}_{ct}) = \frac{1}{n_c n_t} \sum_{i,j=1}^{n} \hat{\mathbf{A}}_{ct}^{ij} = \frac{1}{n_c n_t} \sum_{i,j=1}^{n} \mathbf{A}_{ct}^{ij} \mathbf{W}_{ct}^{ij}, \quad (2b)$$

where $d(\cdot)$ is the density function, n_c and n_t are the numbers of instances in the treatment group and control group. For $d(\hat{\mathbf{A}}_c)$ and $d(\hat{\mathbf{A}}_t)$, the denominators are $n_c n_c$ and $n_t n_t$. τ is the parameter to control the degree of edge density in the learned graph, a larger τ indicates a preference for smaller inter-group density $d(\hat{\mathbf{A}}_{ct})$, and larger intra-group density $d(\hat{\mathbf{A}}_c)$ and $d(\hat{\mathbf{A}}_t)$, and vice versa. By minimizing the $\mathcal{L}_d$ with $\tau > 2$, we can ensure that the learned graph structure adheres to our stated assumption, resulting in lower inter-group edge density compared to intra-group edge density. This provides guidance for cleaning the perturbed graph structure by enabling the weight matrix to assign smaller weights to edges that do not meet the specified condition (typically noise edges), thus effectively denoising the graph structure. Subsequently, we combine the three subgraphs to form the complete graph structure. The adjacency matrices $\hat{\mathbf{A}}_c$, $\hat{\mathbf{A}}_t$ and $\hat{\mathbf{A}}_{ct}$ are mutual orthogonality. Therefore, we can directly sum them up to obtain the overall weighted graph structure $\hat{\mathbf{A}}$.

Exploring Graph Properties

In this section, we aim to ensure that the learned graph structure adheres to the inherent properties of graphs, such as symmetry and feature smoothness. Previous work (Jin et al. 2020) has demonstrated that perturbed edges violate these properties. Consequently, these properties can serve as a reference for cleaning the perturbed graph structure. We adopt $\mathcal{S}$ to capture the graph structure symmetry:

$$\mathcal{S} = \|\hat{\mathbf{A}} - \hat{\mathbf{A}}^T\|_F. \quad (3)$$

Here we employed Equation (3) to capture the graph structure symmetry instead of using the simplistic approach of $\hat{\mathbf{A}}' = (\hat{\mathbf{A}} + \hat{\mathbf{A}}^T)/2$. This enables the model to autonomously learn a graph structure that more accurately reflects the inherent properties of the graph. Feature smoothness suggests that nodes with similar features are more likely to be connected by edges. We capture the graph feature smoothness by adopting:

$$\mathcal{F} = \frac{1}{2} \sum_{i,j=1}^{n} \hat{\mathbf{A}}'_{ij} (\frac{\mathbf{x}_i}{\sqrt{d_i}} - \frac{\mathbf{x}_j}{\sqrt{d_j}})^2, \quad (4)$$

where d_i denotes the degree of nodes i and $(\mathbf{x}_i/\sqrt{d_i} - \mathbf{x}_j/\sqrt{d_j})^2$ measures the features difference between instance i and j. The smaller the $\mathcal{F}$ is, the higher the similarity between connected instances. To ensure the feature smoothness in the weighted graph structure, we should minimize $\mathcal{F}$. The loss function for Exploring Graph Properties can be expressed as:

$$\mathcal{L}_g = \beta \mathcal{S} + \lambda \mathcal{F}. \quad (5)$$

Representation Learning and Outcome Prediction

Then, we use the learned graph structure $\hat{\mathbf{A}}$ and the original feature matrix $\mathbf{X}$ to learn the representation of confounders. In this paper, we employ GCN (Kipf and Welling 2017; Defferrard, Bresson, and Vandergheynst 2016) to to obtain these representations. GCN maps the feature matrix $\mathbf{X}$ and the learned adjacency matrix $\hat{\mathbf{A}}$ into the d-dimensional latent space, which can be formulated as $\mathbf{R} = g(\theta, \hat{\mathbf{A}}, \mathbf{X})$, where θ is the parameters of the network g, and the representation $\mathbf{R} = [\mathbf{r}_1, ..., \mathbf{r}_n] \in \mathbb{R}^{n \times d}$, $\mathbf{r}_i$ denote the representation of the i-th instance. Similar to previous methods in estimating causal effects (Shalit, Johansson, and Sontag 2017; Guo, Li, and Liu 2020), we minimize the Integral Probability Metric (IPM) to balance the distributional divergence of confounders' representations between the treatment and the control groups. We implement IPM with the Wasserstein distance:

$$\mathcal{L}_b = \inf_{g \in \mathcal{G}} \int_{\mathcal{R}} \|g(r) - r\| \, P(r) dr, \quad (6)$$

where $\mathcal{R}$ is the latent space of the representation, $\mathcal{G} = \{g \colon \mathcal{R} \to \mathcal{R} \; s.t. Q(g(r)) = P(r)\}$ denotes the set of functions that map the treated representation distribution $P(r)$ to the controlled representation distribution $Q(r)$.

Then, we use the representation $\mathbf{R}$ to estimate the potential outcome $\mathbf{Y}$ under different treatments $\mathbf{T}$. The output function f can be denoted as $\hat{y}_i^t = f(\theta, \mathbf{r}_i, t)$, where $\mathbf{r}_i$ is the representation of instances i, $\hat{y}_i^t$ is the predicted potential outcome of instance i under the treatment $t \in \{0, 1\}$. We use $\mathcal{L}_p$ to minimize the prediction error:

$$\mathcal{L}_p = \frac{1}{n} \sum_{i=1}^{N} (\hat{y}_i - y_i)^2, \quad (7)$$

where $\hat{y}$ is the predicted outcome and y is the ground truth.

Based on the aforementioned content, the final loss function of EDITOR is given as follows:

$$\mathcal{L} = \mathcal{L}_p + \alpha \mathcal{L}_b + \gamma \mathcal{L}_d + \mathcal{L}_g. \quad (8)$$

The function $\mathcal{L}$ jointly optimizes the weight matrix and the GCN model for the causal effect estimation task. In addition to $\mathcal{L}_d$, and $\mathcal{L}_g$, the information from $\mathcal{L}_p$ and $\mathcal{L}_b$ is also utilized to optimize the weight matrix. This approach ensures that the learned graph structure is more suitable for causal effect estimation tasks, thereby enhancing the effectiveness and robustness of causal effect estimation.

Experiment

In this section, we validate EDITOR's superiority through experiments and comparisons with baseline methods. We examine the reasons for its effectiveness and analyze the impact of its components on performance.

Datasets

Since the counterfactual outcomes are hard to obtain. We follow the standard practice in existing literature (Guo, Li, and Liu 2020; Jiang and Sun 2022) to generate treatments

Table 1: Description of Dataset BlogCatalog and Flickr.

Dataset	Nodes	Features	Edges	ATE Mean	ATE Std
BlogCatalog	5,196	8,189	173,468	22.205	6.416
Flickr	7,575	12,047	293,738	13.471	0.578

and outcomes. In this paper, we use two real-world social network datasets, BlogCatalog (BC) and Flickr (Guo, Li, and Liu 2020; Ma et al. 2021). BlogCatalog is an online community for publishing blogs. In this dataset, each node represents a user, and each edge represents the social relationship between two users. The features are represented by the bag-of-words representation of the bloggers' personal descriptions. Flickr is an online platform for sharing photos and videos. Similarly, in this dataset, each node represents a user, and each edge represents a social relationship. The features of each user represent a list of tags of interest. For each dataset, the synthesis procedure is randomly repeated 10 times, and we report the average results along with their standard deviations. This ensures robustness and reliability in our findings. In Table 1, we present a summary of the statistics for the datasets discussed in this subsection. The average and standard deviation of the ATEs are computed over 10 synthetic iterations.

In this paper, we employ three approaches to perturb the network structure: random edge addition noise, random edge removal noise, and random edge flip noise. We set different perturbation rates to observe the trend of the model's performance variation. If the perturbation rate is greater than 1, for example, the maximum perturbation rate is 5 in Figure 4, it indicates that the number of edges we have modified is five times the number of edges in the original graph. Note that for the edge removal perturbation, the maximum perturbation rate is 1, which implies removing all edges in the graph.

Baseline Methods and Experimental Settings

We conduct experiments to verify the effectiveness of EDITOR, we compare it with ten representative baselines. CFR (Shalit, Johansson, and Sontag 2017), TARNet (Shalit, Johansson, and Sontag 2017),CEVAE (Louizos et al. 2017), DeR-CFR (Wu et al. 2023), DESCN (Zhong et al. 2022) and ESCFR (Wang et al. 2024) are all well-established and effective methods for estimating causal effects. However, these methods are not originally designed for networked observational data and therefore cannot directly utilize the network information. To ensure fairness, we address this limitation by concatenating the corresponding row of the adjacency matrix with the original features, enabling the baselines to leverage network information. ND (Guo, Li, and Liu 2020) and NetEst (Jiang and Sun 2022) are the methods that utilize networked observational data to estimate causal effects. Additionally, we also compare our method with graph structural denoising approaches to demonstrate its effectiveness in causal effect estimation tasks. Pro-GNN (Jin et al. 2020) and RS-GNN (Dai et al. 2022) are two effective graph denoising methods. We employ each of these methods independently to learn the clean graph structure and estimate the causal effect using ND, resulting in two baseline models ND (Pro) and ND (RS).

In the following experiments, we randomly sample 60% and 20% of the instances as the training set and validation set, and use the remaining 20% to be the test set. The hyper-parameters are set as follows: $\alpha = 10^{-3}$, $\beta = 5$, $\gamma = 10^{3}$, $\lambda = 10^{-1}$. For comparison, we employ the baseline methods with their default hyper-parameter settings to ensure a fair evaluation.

Performance Evaluation

In this subsection, we evaluate the performance of EDITOR on BlogCatalog and Flickr datasets and compare it with baseline methods. In our experiments, we utilized two commonly used evaluation metrics in causal inference to assess the performance of our model: the mean absolute error of ATEs and the square root of precision in estimation of heterogeneous effect (PEHE), which are defined as follows:

$$\epsilon ATE = |\frac{1}{n}\sum_{i=1}(\widehat{ITE}_i) - \frac{1}{n}\sum_{i=1}(ITE_i)|, \tag{9}$$

$$\sqrt{\epsilon PEHE} = \sqrt{\frac{1}{n}\sum_{i=1}(\widehat{ITE}_i - ITE_i)^2}, \tag{10}$$

where $\widehat{ITE}_i = \hat{y}_i^1 - \hat{y}_i^0$ and $ITE_i = y_i^1 - y_i^0$ denote the predicted ITE and the ground truth of ITE, respectively. Smaller values of ϵATE and $\sqrt{\epsilon PEHE}$ indicate better model performance, and vice versa.

The results are shown in Tables 2. For random edge addition and edge flip noise, our method significantly outperforms other baseline methods across various perturbation rates. This demonstrates that our EDITOR method has stronger robustness and can effectively recover the clean graph structure from perturbed network data while preserving the underlying causal information. For random edge removal noise, our method also outperforms other baseline methods, although the magnitude of improvement gradually diminishes as the perturbation rate increases. This could be due to the reduction in the number of edges in the graph caused by random edge removal noise. Since our method denoises the graph structure by learning the weights of the edges, a higher perturbation rate results in fewer edges available for weighting. Consequently, the performance improvement of our method decreases as the perturbation rate increases.

Additionally, to more intuitively verify the denoising effect of our method, we show the discrepancy between the graph structures obtained through our method and the original clean graph structures, and compare it with the noisy graph structures. We used the Frobenius norm to calculate the difference. The results are shown in Figure 3. As shown, the difference between our method's graph structures and the clean graph structures decreases as training progresses, demonstrating the effectiveness of our method for graph denoising. This indicates that our approach effectively removes noisy edges during training, resulting in a graph structure that more closely approximates the original clean graph structure. It is worth noting that the improvement effect of

Table 2: Performance comparison on BlogCatalog and Flickr datasets under different perturbation types and perturbation rates.

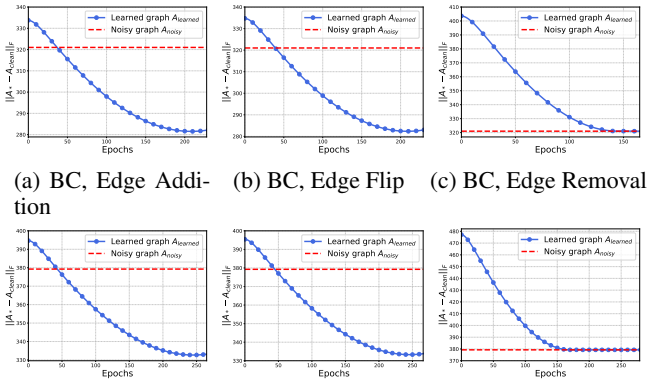
Perturbation Type	Dataset	BlogCatalog						Flickr					
	Perturbation Rate	0.2		0.5		1		0.2		0.5		1	
	Method	$\sqrt{\epsilon PEHE}$	ϵATE	$\sqrt{\epsilon PEHE}$	ϵATE	$\sqrt{\epsilon PEHE}$	ϵATE	$\sqrt{\epsilon PEHE}$	ϵATE	$\sqrt{\epsilon PEHE}$	ϵATE	$\sqrt{\epsilon PEHE}$	ϵATE
Edge Addition	CFR	27.50±10.89	16.83±7.98	27.92±10.66	17.38±7.64	28.06±10.43	17.67±7.25	27.37±0.87	7.14±1.35	27.56±1.05	7.37±1.24	27.65±1.07	7.59±1.21
	TARNet	27.68±10.86	17.19±7.77	28.14±10.58	17.81±7.42	28.28±10.28	17.99±6.97	27.27±0.66	7.10±1.32	27.28±0.68	7.27±1.31	27.64±0.85	7.41±1.26
	CEVAE	21.93±2.59	8.44±1.73	21.83±2.46	8.18±1.48	22.44±2.57	9.71±1.62	21.33±0.41	4.31±0.93	23.41±1.825	8.68±2.84	21.22±0.36	4.16±0.65
	DESCN	19.32±4.92	5.95±7.21	19.88±4.91	5.93±7.24	19.83±4.93	6.07±7.34	26.74±1.62	2.53±1.51	26.57±1.64	2.58±1.20	26.58±1.62	1.88±1.26
	DeR-CFR	21.91±0.59	16.46±1.52	22.20±0.41	16.47±1.73	22.18±0.29	16.37±1.93	27.34±1.32	11.78±0.29	27.38±1.32	11.65±0.32	27.34±1.38	11.34±0.28
	ESCFR	16.18±2.25	3.17±1.95	17.14±3.15	2.76±2.13	19.14±4.71	3.75±4.40	22.48±8.99	3.34± 1.94	24.58±7.78	2.12± 0.81	24.92±3.46	2.71± 1.59
	NetEst	17.59±9.54	5.05±4.97	18.88±9.93	6.44±5.78	19.88±9.78	5.89±4.38	20.09±3.09	2.66±1.40	22.60±3.15	**1.96±1.36**	23.46±2.96	2.06±1.20
	ND	9.86±1.92	3.52±1.10	11.71±3.66	2.31±1.79	12.33±3.08	3.56±3.31	11.28±3.10	2.97±2.96	12.32±2.39	3.30±2.48	13.25±2.57	4.68±2.64
	ND(RS)	11.75±3.80	2.35±1.37	11.59±3.26	3.24±1.29	11.70±3.04	2.97±1.99	8.65±1.99	2.72±1.02	8.46±2.13	2.78±1.39	8.77±2.07	3.16±1.15
	ND(Pro)	8.85±2.49	3.08±2.78	9.27±2.23	4.06±1.94	10.20±2.92	4.41±2.26	8.17±1.94	3.14±1.27	8.06±1.73	3.01±1.34	8.33±1.94	3.25±1.45
	EDITOR(ours)	**8.74±2.30**	**2.26±2.09**	**8.90±2.04**	**1.74±1.81**	**9.01±2.21**	**1.63±1.69**	**7.37±1.58**	**1.97±0.90**	**7.59±1.46**	2.05±1.15	**7.58±1.42**	**1.84±1.04**
Edge Removal	CFR	26.24±11.20	14.91±8.29	25.09±10.93	12.69±7.68	20.49±6.55	**3.04±2.60**	26.93±1.05	6.56±1.09	26.60±1.44	5.52±1.27	27.14±0.84	1.94±1.06
	TARNet	26.28±11.44	15.03±8.34	25.15±11.03	12.94±7.56	20.17±6.52	3.19±2.47	26.86±0.78	6.47±1.16	26.51±1.37	5.59±1.25	26.96±0.74	**1.51±1.13**
	CEVAE	22.19±2.69	8.89±1.97	21.62±2.58	7.44±1.81	22.44±2.65	9.57±1.89	21.54±0.31	5.47±0.78	21.30±0.34	4.50±0.74	**21.02±0.28**	3.28±0.47
	DESCN	20.07±4.87	5.89±7.24	20.35±4.64	6.11±7.27	20.17±5.01	6.14±6.98	26.92±1.64	2.01±1.19	26.94±1.44	2.58±1.09	26.87±1.67	1.99±1.61
	DeR-CFR	21.99±1.09	16.42±1.10	22.09±0.84	16.48±1.41	21.33±1.44	15.46±0.44	27.41±1.31	11.99±0.27	27.38±1.36	12.02±0.20	27.41±1.33	11.55±0.16
	ESCFR	18.34±3.97	2.69±2.15	20.04±4.64	5.28±10.35	28.20±8.55	5.14±1.59	11.04±6.18	3.78±1.43	11.14±6.10	3.08± 1.58	32.65±1.91	1.81±1.54
	NetEst	17.18±9.55	5.08±5.29	17.52±9.09	4.46±4.11	22.10±7.84	4.17±3.29	22.32±3.21	2.42±1.62	22.43±3.13	1.96±1.35	31.49±2.90	4.12±2.83
	ND	10.71±2.63	3.70±2.49	11.22±3.19	3.16±1.89	23.33±9.44	7.42±5.59	8.72±1.81	2.98±2.08	7.93±1.51	2.28±1.08	26.20±3.12	1.73±1.82
	ND(RS)	12.06±3.07	3.59±2.41	13.60±5.17	2.99±1.58	23.33±9.35	7.85±5.82	10.23±2.34	2.82±1.28	11.53±2.59	2.37±1.15	26.42±1.67	2.05±1.65
	ND(Pro)	9.54±2.23	3.63±2.14	9.99±2.12	3.39±2.29	23.54±9.38	7.55±5.70	7.97±1.77	3.06±1.08	8.29±1.91	3.20±1.19	25.79±2.22	1.80±1.49
	EDITOR(ours)	**9.34±2.02**	**2.65±1.64**	**9.98±2.13**	**2.57±1.05**	**19.35±8.44**	4.73±4.98	**7.29±1.59**	**1.97±0.99**	**7.32±1.52**	**1.96±1.06**	24.10±1.39	1.90±1.69
Edge Flip	CFR	27.49±10.96	16.81±7.98	27.85±10.71	17.34±7.69	28.04±10.43	17.67±7.24	27.39±0.85	7.11±1.30	27.58±0.99	7.36±1.23	27.74±0.89	7.59±1.16
	TARNet	27.66±10.90	17.11±7.88	28.12±10.56	17.78±7.41	28.21±10.29	17.93±6.97	27.22±0.61	7.05±1.44	27.36±0.71	7.33±1.47	27.64±0.91	7.49±1.34
	CEVAE	22.12±2.69	8.82±1.98	22.25±2.69	9.19±1.98	22.05±2.66	8.77±1.87	21.06±0.27	3.49±0.39	21.16±0.31	3.86±0.65	21.12±0.29	3.68±0.59
	DESCN	19.98±4.93	5.96±7.22	19.91±4.89	5.93±7.25	19.84±4.93	6.02±7.37	26.74±1.67	1.96±1.18	26.55±1.63	1.57±1.20	26.62±1.64	1.90±1.24
	DeR-CFR	22.04±0.64	16.61±1.72	22.20±0.36	16.54±1.87	22.12±0.34	16.28±1.84	27.44±1.32	11.91±0.17	27.38±1.41	11.55±0.39	27.37±1.38	11.36±0.19
	ESCFR	16.46±3.44	3.13±2.83	17.15±2.20	2.86±2.10	18.98±4.20	6.90±13.74	20.15±8.07	2.92±1.30	24.33±6.74	1.81± 0.76	25.39±3.35	2.69± 0.97
	NetEst	17.82±9.59	5.65±4.98	18.93±9.99	6.89±5.38	19.76±9.38	5.55±3.75	22.37±3.01	2.16±1.38	22.69±2.99	1.77±1.19	23.47±2.88	2.20±1.43
	ND	11.34±3.35	**2.16±1.63**	12.93±4.93	5.41±5.40	14.67±8.36	6.65±8.67	12.70±2.95	4.49±2.59	13.03±2.58	4.27±2.01	13.93±3.11	4.68±3.72
	ND(RS)	11.94±3.59	3.25±2.31	11.60±2.98	3.09±2.15	11.78±3.20	3.33±1.43	8.83±1.99	2.90±1.17	8.72±2.11	2.87±1.20	8.65±1.87	2.93±1.38
	ND(Pro)	9.21±2.44	3.40±2.11	8.87±2.87	3.03±2.69	10.29±3.41	3.73±2.39	8.06±1.91	3.06±1.39	9.04±3.26	3.86±2.30	8.30±2.02	3.26±1.67
	EDITOR(ours)	**9.00±2.05**	2.28±1.58	**9.45±2.34**	**2.11±2.05**	**9.61±2.58**	**2.21±1.83**	**7.19±1.49**	**1.86±0.76**	**7.27±1.42**	**1.48±0.67**	**7.54±1.29**	**1.86±1.08**

our method is not significant under removal perturbations, which is consistent with our previous observations. Since our method adopts edge weighting for denoising, it may not be as effective for edge removal perturbations. However, as seen from the experiments in Table 2, our accuracy is still improved, proving that we can still obtain sufficient information from the remaining edges in the graph to enhance the model's performance.

Ablation Studies

In this subsection, we conducted ablation studies to investigate the impact of each component on our model's performance. Firstly, we explored the effects of the edge weights matrices $\mathbf{W}_{ct}$, $\mathbf{W}_c$, and $\mathbf{W}_t$. Based on the original model, we made modifications to create three new models: EDITOR-$\mathbf{W}_{ct}$, EDITOR-$\mathbf{W}_c$, and EDITOR-$\mathbf{W}_t$. During the training process of model EDITOR-$\mathbf{W}_{ct}$, the parameters of $\mathbf{W}_c$ and $\mathbf{W}_t$ are fixed, only update the parameters of matrix $\mathbf{W}_{ct}$. The models EDITOR-$\mathbf{W}_c$ and EDITOR-$\mathbf{W}_t$ follow the same principle. Figures 4(a) - 4(f) present the ablation results for models where only one weights matrix is updated. It can be seen that there is a significant decline in performance for all these models compared to EDITOR. This demonstrates the essential impact of the edge weights matrices on improving the model's performance, the combination of the three matrices facilitates a better capturing of the causal information in the network.

Additionally, our method also includes three important components: edge density control, symmetry, and feature smoothness. We created four model variants to observe the effects of these components. The EDITOR-Base indicates no constraints applied, while EDITOR-w/Dens, EDITOR-w/Sym, and EDITOR-w/FS represent models with edge density control, symmetry, and feature smoothness constraints applied on top of the EDITOR-Base model, respec-



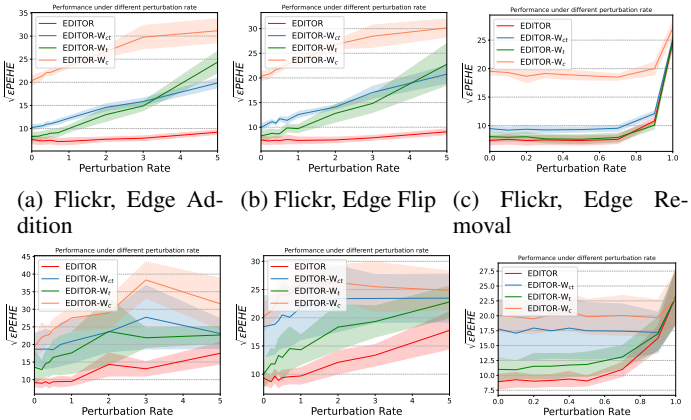
(a) BC, Edge Addition (b) BC, Edge Flip (c) BC, Edge Removal

(d) Flickr, Edge Addition (e) Flickr, Edge Flip (f) Flickr, Edge Removal

Figure 3: The discrepancy between the learned graph structure and the original clean graph structure, with red dashed lines indicating the perturbed noisy graph.

Table 3: Result of ablation studies for different constraints applied.

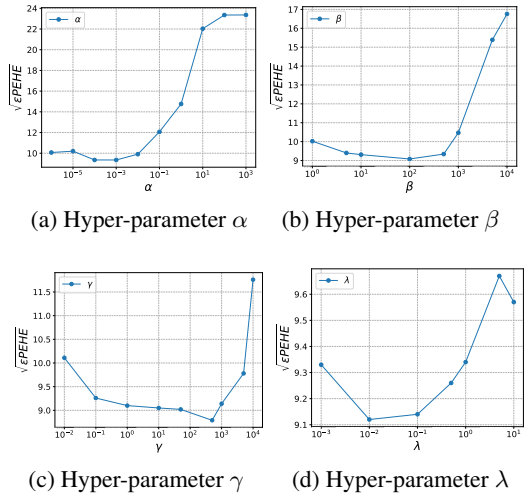
Perturbation Type	Perturbation Rate	Result of $\sqrt{\epsilon \text{PEHE}}$				
		EDITOR-Base	EDITOR-w/Sym	EDITOR-w/Dens	EDITOR-w/FS	EDITOR
Edge Addition	0.05	8.29±1.84	8.20±1.83	8.10±1.80	7.66±1.82	**7.46±1.60**
	0.3	8.51±1.84	8.31±1.70	8.17±1.93	7.35±1.44	**7.26±1.46**
	0.7	8.41±1.77	8.32±1.84	8.27±1.87	**7.34±1.32**	7.35±1.35
	1	8.43±1.57	8.30±1.86	8.54±1.66	7.96±2.65	**7.31±1.51**
Edge Removal	0.05	8.25±1.93	8.16±1.95	7.91±1.92	7.40±1.57	**7.23±1.58**
	0.3	7.82±1.81	7.84±1.83	7.85±1.88	7.51±1.68	**7.41±1.65**
	0.7	8.48±2.06	8.39±1.88	8.46±2.11	7.88±1.77	**7.73±1.50**
	1	25.18±1.51	24.37±1.63	24.33±1.58	24.49±1.61	**24.10±1.39**
Edge Flip	0.05	8.17±1.85	8.02±1.79	7.97±1.79	7.41±1.60	**7.29±1.47**
	0.3	8.38±1.86	8.37±1.74	8.06±1.71	7.28±1.47	**7.10±1.46**
	0.7	8.55±1.70	8.53±1.99	8.28±1.79	7.31±1.17	**7.20±1.46**
	1	8.19±1.81	8.29±1.70	8.23±1.83	7.32±1.38	**7.29±1.40**



(a) Flickr, Edge Addition (b) Flickr, Edge Flip (c) Flickr, Edge Removal

(d) BC, Edge Addition (e) BC, Edge Flip (f) BC, Edge Removal

Figure 4: Result of ablation studies for edge weights matrices, with red line indicating our method.



(a) Hyper-parameter α (b) Hyper-parameter β

(c) Hyper-parameter γ (d) Hyper-parameter λ

Figure 5: Result of parameter analysis on BlogCatalog.

tively. The results as shown in Table 3, it can be observed that compared to EDITOR-Base, the three models all show varying degrees of improvement in performance, demonstrating the effectiveness of the three constraints for graph structure denoising. The combination of these constraints results in the EDITOR model achieving the best overall performance.

Parameter Analysis

In this subsection, we investigated the impact of hyper-parameters on the performance of EDITOR. We fixed the values of other hyper-parameters as described in Section and varied the parameters α, β, γ, and λ individually. The experimental results are shown in Figure 5. Due to space limitations, we report the results for the BlogCatalog dataset with a perturbation rate set to 0.2. The perturbation type is edge flips, which includes both edge additions and removals, making it more diverse and convincing. As seen in Figure 5, when the parameter values are within a certain range, EDITOR achieves its best performance. However, if the parameter values are set too low, the performance of EDITOR decreases, indicating that all four loss functions contribute to improving the model's performance. Conversely, setting the parameter values too high leads to a decline in performance, suggesting that focusing too heavily on a specific component can reduce the model's overall performance.

Conclusion

When the graph structure in networked observational data contains noise or is subjected to perturbations, the performance of current causal effect estimation methods significantly decreases. To address this problem, we proposed EDITOR, a method designed to eliminate graph structure noise and restore causal information in the network. EDITOR achieves this by partitioning the graph into distinct subgraphs based on edge types and learning their importance weights respectively. Extensive experiments demonstrated the effectiveness and robustness of EDITOR.

References

Atwood, J.; and Towsley, D. 2016. Diffusion-convolutional neural networks. *Advances in neural information processing systems*, 29.

Cheng, L.; Guo, R.; Moraffah, R.; Candan, K. S.; Raglin, A.; and Liu, H. 2020. A practical data repository for causal learning with big data. In *Benchmarking, Measuring, and Optimizing: Second BenchCouncil International Symposium, Bench 2019, Denver, CO, USA, November 14–16, 2019, Revised Selected Papers 2*, 234–248. Springer.

Chu, Z.; and Li, S. 2023. Causal effect estimation: Recent progress, challenges, and opportunities. *Machine Learning for Causal Inference*, 79–100.

Chu, Z.; Rathbun, S. L.; and Li, S. 2021. Graph infomax adversarial learning for treatment effect estimation with networked observational data. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, 176–184.

Dai, E.; Jin, W.; Liu, H.; and Wang, S. 2022. Towards Robust Graph Neural Networks for Noisy Graphs with Sparse Labels. In *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining*, WSDM '22, 181–191. New York, NY, USA: Association for Computing Machinery. ISBN 9781450391320.

Dai, H.; Li, H.; Tian, T.; Huang, X.; Wang, L.; Zhu, J.; and Song, L. 2018. Adversarial attack on graph structured data. In *International conference on machine learning*, 1115–1124. PMLR.

Defferrard, M.; Bresson, X.; and Vandergheynst, P. 2016. Convolutional neural networks on graphs with fast localized spectral filtering. *Advances in neural information processing systems*, 29.

Entezari, N.; Al-Sayouri, S. A.; Darvishzadeh, A.; and Papalexakis, E. E. 2020. All you need is low (rank) defending against adversarial attacks on graphs. In *Proceedings of the 13th International Conference on Web Search and Data Mining*, 169–177.

Gao, C.; Zheng, Y.; Wang, W.; Feng, F.; He, X.; and Li, Y. 2024. Causal inference in recommender systems: A survey and future directions. *ACM Transactions on Information Systems*, 42(4): 1–32.

Gui, H.; Xu, Y.; Bhasin, A.; and Han, J. 2015. Network a/b testing: From sampling to estimation. In *Proceedings of the 24th International Conference on World Wide Web*, 399–409.

Guo, R.; Cheng, L.; Li, J.; Hahn, P. R.; and Liu, H. 2020. A survey of learning causality with data: Problems and methods. *ACM Computing Surveys (CSUR)*, 53(4): 1–37.

Guo, R.; Li, J.; and Liu, H. 2020. Learning individual causal effects from networked observational data. In *Proceedings of the 13th international conference on web search and data mining*, 232–240.

Huang, Q.; Ma, J.; Li, J.; Sun, H.; and Chang, Y. 2022a. SemiITE: Semi-supervised Individual Treatment Effect Estimation via Disagreement-Based Co-training. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, 400–417. Springer.

Huang, Q.; Yamada, M.; Tian, Y.; Singh, D.; and Chang, Y. 2022b. Graphlime: Local interpretable model explanations for graph neural networks. *IEEE Transactions on Knowledge and Data Engineering*.

Hünermund, P.; and Bareinboim, E. 2023. Causal inference and data fusion in econometrics. *The Econometrics Journal*, utad008.

Jiang, S.; and Sun, Y. 2022. Estimating Causal Effects on Networked Observational Data via Representation Learning. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, 852–861.

Jin, W.; Li, Y.; Xu, H.; Wang, Y.; Ji, S.; Aggarwal, C.; and Tang, J. 2021. Adversarial Attacks and Defenses on Graphs. *SIGKDD Explor. Newsl.*, 22(2): 19–34.

Jin, W.; Ma, Y.; Liu, X.; Tang, X.; Wang, S.; and Tang, J. 2020. Graph structure learning for robust graph neural networks. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, 66–74.

Johansson, F.; Shalit, U.; and Sontag, D. 2016. Learning representations for counterfactual inference. In *International conference on machine learning*, 3020–3029. PMLR.

Kipf, T. N.; and Welling, M. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *International Conference on Learning Representations*.

Louizos, C.; Shalit, U.; Mooij, J. M.; Sontag, D.; Zemel, R.; and Welling, M. 2017. Causal effect inference with deep latent-variable models. *Advances in neural information processing systems*, 30.

Ma, J.; Guo, R.; Chen, C.; Zhang, A.; and Li, J. 2021. Deconfounding with networked observational data in a dynamic environment. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*, 166–174.

Ma, J.; and Li, J. 2022. Learning causality with graphs. *AI Magazine*, 43(4): 365–375.

Ma, Y.; and Tresp, V. 2021. Causal inference under networked interference and intervention policy enhancement. In *International Conference on Artificial Intelligence and Statistics*, 3700–3708. PMLR.

McPherson, M.; Smith-Lovin, L.; and Cook, J. M. 2001. Birds of a feather: Homophily in social networks. *Annual review of sociology*, 27(1): 415–444.

Prosperi, M.; Guo, Y.; Sperrin, M.; Koopman, J. S.; Min, J. S.; He, X.; Rich, S.; Wang, M.; Buchan, I. E.; and Bian, J. 2020. Causal inference and counterfactual prediction in machine learning for actionable healthcare. *Nature Machine Intelligence*, 2(7): 369–375.

Schlichtkrull, M.; Kipf, T. N.; Bloem, P.; Van Den Berg, R.; Titov, I.; and Welling, M. 2018. Modeling relational data with graph convolutional networks. In *The Semantic Web: 15th International Conference, ESWC 2018, Heraklion, Crete, Greece, June 3–7, 2018, Proceedings 15*, 593–607. Springer.

Shalit, U.; Johansson, F. D.; and Sontag, D. 2017. Estimating individual treatment effect: generalization bounds and

algorithms. In *International conference on machine learning*, 3076–3085. PMLR.

Vaughan, L.; Kipp, M. E.; and Gao, Y. 2007. Why are websites co-linked? The case of Canadian universities. *Scientometrics*, 72: 81–92.

Veitch, V.; Wang, Y.; and Blei, D. 2019. Using embeddings to correct for unobserved confounding in networks. *Advances in Neural Information Processing Systems*, 32.

Veličković, P.; Cucurull, G.; Casanova, A.; Romero, A.; Liò, P.; and Bengio, Y. 2018. Graph Attention Networks. In *International Conference on Learning Representations*.

Wang, H.; Fan, J.; Chen, Z.; Li, H.; Liu, W.; Liu, T.; Dai, Q.; Wang, Y.; Dong, Z.; and Tang, R. 2024. Optimal transport for treatment effect estimation. *Advances in Neural Information Processing Systems*, 36.

Wu, A.; Yuan, J.; Kuang, K.; Li, B.; Wu, R.; Zhu, Q.; Zhuang, Y.; and Wu, F. 2023. Learning Decomposed Representations for Treatment Effect Estimation. *IEEE Transactions on Knowledge and Data Engineering*, 35(5): 4989–5001.

Wu, H.; Wang, C.; Tyshetskiy, Y.; Docherty, A.; Lu, K.; and Zhu, L. 2019. Adversarial examples for graph data: deep insights into attack and defense. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, 4816–4823.

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

Ying, R.; He, R.; Chen, K.; Eksombatchai, P.; Hamilton, W. L.; and Leskovec, J. 2018. Graph convolutional neural networks for web-scale recommender systems. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, 974–983.

Yuan, Y.; Altenburger, K.; and Kooti, F. 2021. Causal network motifs: identifying heterogeneous spillover effects in A/B tests. In *Proceedings of the Web Conference 2021*, 3359–3370.

Zeng, J.; and Wang, R. 2022. A survey of causal inference frameworks. *arXiv preprint arXiv:2209.00869*.

Zhong, K.; Xiao, F.; Ren, Y.; Liang, Y.; Yao, W.; Yang, X.; and Cen, L. 2022. DESCN: Deep entire space cross networks for individual treatment effect estimation. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 4612–4620.

Zügner, D.; Akbarnejad, A.; and Günnemann, S. 2018. Adversarial attacks on neural networks for graph data. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, 2847–2856.