Estimating Causal Effects on Networked Observational Data via Representation Learning

Song Jiang
University of California, Los Angeles
Los Angeles, CA
songjiang@cs.ucla.edu

Yizhou Sun
University of California, Los Angeles
Los Angeles, CA
yzsun@cs.ucla.edu

ABSTRACT

In this paper, we study the causal effects estimation problem on networked observational data. We theoretically prove that standard graph machine learning (ML) models, e.g., graph neural networks (GNNs), fail in estimating the causal effects on networks. We show that graph ML models exhibit two distribution mismatches of their objective functions compared to causal effects estimation, leading to the failure of traditional ML models. Motivated by this, we first formulate the networked causal effects estimation as a data-driven multi-task learning problem, and then propose a novel framework **NetEst** to conduct causal inference in the network setting. NetEst uses GNNs to learn representations for confounders, which are from both a unit's own characteristics and the network effects. The embeddings are then used to sufficiently bridge the distribution gaps via adversarial learning and estimate the observed outcomes simultaneously. Extensive experimental studies on two real-world networks with semi-synthetic data demonstrate the effectiveness of NetEst. We also provide analyses on why and when NetEst works.

CCS CONCEPTS

• **Mathematics of computing → Causal networks**.

KEYWORDS

causal inference, network effect, graph neural networks

ACM Reference Format:
Song Jiang and Yizhou Sun. 2022. Estimating Causal Effects on Networked Observational Data via Representation Learning. In *Proceedings of the 31st ACM Int'l Conference on Information and Knowledge Management (CIKM '22), Oct. 17–21, 2022, Atlanta, GA, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3511808.3557311

1 INTRODUCTION

Causal inference (also formalized as counterfactual reasoning [9, 33]) has attracted increasing interests on networked scenarios, such as social networks [30, 31], online advertisements [29], and vaccine distribution [4]. Randomized controlled trials (RCTs) are still the "gold standard" on networked data [11, 45]. However, RCTs are usually time-consuming, highly-costly and even not doable, which is especially true in the context of networks. Therefore, estimating the causal effects from networked observational data is an important yet challenging problem and the focus of this paper. We use vaccine distribution as our motivating example throughout this paper. Given the observed vaccine assignments (i.e., treatment) and the immunity level [27] (i.e., outcome) of a social community (i.e., network), we aim to answer the counterfactual questions like "would the community immunity level be stronger had a different group of people been vaccinated"?

The difficulties of causal inference on networked data are due to the dependency between units in a network and the need of inductive inference raised by real-world applications. First, compared with traditional independent setting [19, 37, 43, 44], the non-i.i.d. nature of networks introduces two-fold challenges to causal inference, i.e., *homophily* [28] and *interference* [18]. Homophily describes the phenomenon that similar units in networks tend to form social ties, which brings in new confounders (factors that affect both treatment and potential outcome) for causal effects in addition to the units' own features (a.k.a., characteristics). Interference refers to the fact that the potential outcome of a unit is caused by not only their own but the neighbors' treatments on networks, e.g., getting vaccines protects both a person and their social contacts. In other words, the traditional SUTVA [35] assumption that one's potential outcome is stable regardless of the treatment assignments of others is no longer valid. Second, from the empirical view, many real-world problems require to predict the causal effects on a *new* network without any observed outcomes (known as "out-of-sample" estimation [37], or "inductive" prediction [15] in machine learning), e.g., finding out the best initial vaccine plan for a community. However, transferring the estimation from the observed networks to a new network is non-trivial because two network structures could be quite distinct.

To estimate causal effects on networked data, Forastiere *et al.* [7] extend the "no unobserved confounders" assumption to networks with interference, and propose a networked propensity score based method to infer the causal effects. Arbour *et al.* [2] find out the adjustment variables on networks and estimate the treatment effects via back-door criterion [33]. Despite the success on networks with observed outcomes (known as "within-sample" estimation [37]), these methods are not able to generalize the effects to a new network where we do not have any outcomes observed. Recent works propose to use network embeddings to capture unobserved confounders encoded in network structure [6, 12, 14, 25, 39]. However, these works still follow the STUVA assumption and ignore the interference, which induces estimation bias of real-world networks.

Given that networked observational data contains features, treatments, observed outcomes and the network structure, a natural idea is to train a standard graph machine learning model, e.g., graph neural networks (GNNs) [23, 40–42], on the observed data and then

CIKM '22, October 17–21, 2022, Atlanta, GA, USA

ACM ISBN 978-1-4503-9236-5/22/10.
https://doi.org/10.1145/3511808.3557311

to predict the counterfactual outcomes for causal effects estimation. However, we theoretically demonstrate that such standard graph machine learning models fail in inferring the causal effects on networks, because there are two distribution mismatches between their objective functions (details in Sec. 3.1). In other words, standard graph machine learning models are solving a different optimization goal from estimating the causal effects on networks. To fill this gap, we further find that it is sufficient to enforce the two mismatched distributions to be uniform. These insights motivate us to propose a novel framework **NetEst**, which formulates the **Net**worked causal effects **Est**imation into a data-driven multi-task paradigm with two optimization goals: predicting the potential outcomes and bridging the distribution gaps between standard graph machine learning and networked causal inference. To facilitate this, NetEst first uses GNNs to encode the confounders that are from both a unit's own and neighbors' features into latent representations. Together with both a unit's own and their neighbors' treatments, these embeddings are then used to estimate the potential outcomes via an estimator. Meanwhile, NetEst uses two adversarial learning modules to force the mismatched distributions to follow uniform distributions based on the embeddings. NetEst is applicable to both the "out-of-sample" [37] and the traditional "within-sample" [19] estimation on networked data.

Our **main contributions** are summarized as follows: *First*, we theoretically prove that standard graph machine learning models can not estimate causal effects on networks due to the distribution mismatches between their objective functions. *Second*, we formalize the networked causal effects estimation to a multi-task learning problem and propose a novel framework NetEst that solves the distribution gaps and alleviates the challenges induced by the nature of neworked data. *Third*, we conduct extensive experiments on two datasets, demonstrating the effectiveness of NetEst and present empirical analyses of why and when NetEst works.

2 PROBLEM SETUP

We follow Arbour *et al.* [2] to set up the causal effects estimation on networks. We first discuss the causal graph of networked data in the presence of homophily and interference. Then we present the definition of causal effects on networks and discuss its identification. We list all the notations used in this paper in Table 1.

2.1 Causal Graph on Networks

Causal graph is a directed acyclic graph (DAG) that describes the causal relations among variables [33]. Without loss of generality, we still use vaccination as our motivating example to depict a plausible causal graph on networks in Fig. 1. The social structure of a three-unit community is described on the left, and right part shows the causal relations of their features, treatments and potential outcomes. In practice, a unit's features (e.g., health condition) cause both their (1) decisions to get vaccinated (treatment) and (2) immunity to a virus (potential outcome), namely a unit's features contain confounders between treatment and potential outcome (indicated by red edges in Fig. 1). In addition, a unit's features may also affect the neighbors' treatments and potential outcomes as they may influence each other. For example, a person with a weakened immune system may increase their risk of infection, prompting their family members to get vaccinated. In other words, networks introduce new confounders between the treatment and potential outcome (blue edges in Fig. 1). Different from the independent setting, treatment of a unit spills over to their neighbors. For example, getting vaccination protects not only oneself, but others in the community (i.e., herd immunity [1]). This "peer effect" reflects the interference nature of the network (marked by green edges in Fig. 1). Following [2], we assume that network confounders and the peer effect only exist among *1-hop* neighbors. We further assume that all treatments are carried out at the same time without any order. In other words treatments will not affect each other. The readers can refer to [32] for other plausible causal graphs on networked data.

Table 1: Notations used in this paper.

Symbol	Description
G, A, X	graph, adjacency matrix and feature matrix
t_i, x_i, y_i	treatment, feature and potential outcome of unit i
$\{x_j\}_{j\in\mathcal{N}_i}$	features of i's neighbors in network
T, Y	treatment vector, potential outcome vector of all users
$\{t_j\}_{j\in\mathcal{N}_i}$	treatments of i's neighbors in network
$\{x_j\}_{j\in-\mathcal{N}_i}$	treatments of i's non-neighbors in network
Z	summary function of neighbors' treatments
z_i	peer exposure of unit i
$Y^i_{t_i,z_i}$	observed outcome of unit i under t_i and z_i
$Y_i\|do(t_i=t,z_i=z)$	potential outcome of unit i under t_i and z_i
$\mathcal{N}_i, N_i$	i's neighbors set, and size
$\tau, \hat{\tau}$	treatment effects, estimate of treatment effects
ϕ, s_i	representation function, representation of unit i
m	outcome estimation function from representation
f	outcome estimation function from feature
d_t, d_z	discriminators
$\mathcal{J}$	loss function

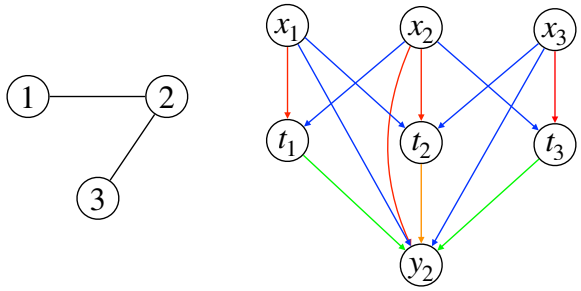

Figure 1: Causal graph of a network with three nodes. Left: the network connection topology. Right: causal graph. x, t, y are features, treatments and potential outcome respectively. A red edge shows confounders from a unit's own features, a blue edge means confounders brought by network, an orange edge represents the causal effects between a node's own treatment and potential outcome, and a green edge shows the peer effects of treatment. Only y_2 is shown for simplicity. We assume that peer effects occur only between 1-hop neighbors and that there are no unmeasured confounders.

2.2 Causal Inference on Networked Data

Formally, given a network $G = \langle A, X \rangle$, in which $A \in \mathbb{R}^{V \times V}$ is the adjacency matrix where V is the number of units (nodes) in G; $X \in \mathbb{R}^{V \times k}$ is the units' features matrix and k is the feature dimension. We use $x_i \in \mathbb{R}^k$ to represent the feature of i-th node. We denote $T = [t_1, ..., t_V]$ as the treatment vector of all V units where $t_i \in \{0, 1\}$ is the treatment of i-th unit. Following many existing works [13, 19, 37, 39], we assume that t_i is binary (e.g., $t_i = 1$ means getting vaccinated while 0 means not). We then denote the potential outcome vector $Y = [y_1, ..., y_V]$ where $y_i \in \mathbb{R}$ is the potential outcome of unit i. We further assume y_i is continuous (e.g., a higher value means a stronger immunity). Following [2], we can define the causal effects $\tau(X)$ on the whole network G as the difference in the potential outcomes under two treatments vectors T' and T'', which is formalized as:

$$\tau(X) \coloneqq \mathbb{E}\left[Y|do(T') - Y|do(T'')\middle|X, A\right], \tag{1}$$

where the *do*-calculus [33] represents an intervention on treatments. In our motivating example, T' and T'' can be two vaccine distribution strategies. With Eq. (1), we can answer causal questions on networks, such as comparing the impacts of two vaccine plans.

To measure the overall effects $\tau(X)$ on the entire network, we need to estimate the treatment effects $\tau(x_i)$ for every unit, namely the individual treatment effects (ITE). From the individual view, a unit's potential outcome y_i is caused by their own feature x_i, treatment t_i, neighbors' features $\{x_j\}_{j \in \mathcal{N}_i} \in \mathbb{R}^{N_i \times k}$ and treatments $\{t_j\}_{j \in \mathcal{N}_i} \in \{0, 1\}^{N_i}$ as in Fig. 1, where N_i is the number of 1-hop neighbors of unit i. To represent the interference of neighbors' treatments $\{t_j\}_{j \in \mathcal{N}_i}$, following [7], we define a summary function $Z : 2^T \to [0, 1]$ that reduces a set of treatments in 2^T into a scalar. We set $z_i = Z(\{t_j\}_{j \in \mathcal{N}_i})$, where z_i is defined as the *peer exposure* of unit i to neighbors' treatments $\{t_j\}_{j \in \mathcal{N}_i}$. In this paper, we define Z as a function to calculate the percentage of treated neighbors, i.e., $z_i = \sum_{j \in \mathcal{N}_i} t_j / |N_i|$, and thus z_i means the ratio of i's neighbors whose treatments are 1. Therefore, the range of z_i is $[0, 1]$. To highlight these causes, we reformulate unit i's potential outcome y_i as $Y_i|do(t_i = t, z_i = z)$, indicating the potential outcome under the treatment t and the peer exposure z. Then individual treatment effects (ITE) $\tau(x_i)$ of unit i can be formalized as:

$$\tau(x_i) \coloneqq \mathbb{E}\left[Y_i|do(t_i = t', z_i = z') - Y_i|do(t_i = t'', z_i = z'')\middle|x_i, \{x_j\}_{j \in \mathcal{N}_i}\right]. \tag{2}$$

Given the treatment vector T and the topology A of networks, we can compute the peer exposure z_i for every unit. Therefore, the network effects can be fully represented by the peer exposure z_i and neighbors' features $\{x_j\}_{j \in \mathcal{N}_i}$ from the individual view. The presence of z_i and $\{x_j\}_{j \in \mathcal{N}_i}$ in Eq. (2) indicates the major difference of networked ITE compared to the general independent scenarios where potential outcomes are not affected by neighbors' treatments.

To estimate ITE $\tau(x_i)$ in Eq. (2), we need the two potential outcomes under different treatments and peer exposures. However, we can only observe at most one of them from observational data. For instance, we can only observe the outcomes of a community w.r.t. one vaccine distribution plan. Therefore, the core of $\tau(x_i)$ is to estimate the counterfactual outcome, namely the treatment-peer exposure-potential outcome tuples that are not observed.

Eq. (2) enables us to further study some interesting causal effects questions on networks. As in [2], we focus the following three:

- *Individual effects:* $Y_i|do(t_i = 1, z_i = 0) - Y_i|do(t_i = 0, z_i = 0)$. It represents unit i's own treatment effects, e.g., how much protection would I get if it was just me and none of my friends were vaccinated?
- *Peer effects:* $Y_i|do(t_i = 0, z_i = z') - Y_i|do(t_i = 0, z_i = z'')$. It reflects the effects of treatment inference, e.g., how much protection would I get if different groups of my friends but not me were vaccinated?
- *total effects:* $Y_i|do(t_i = 1, z_i = 1) - Y_i|do(t_i = 0, z_i = 0)$. It describes the combined effects of individual treatment and the network interference, e.g., how much protect would I get if everyone is vaccinated?

2.3 Causal Identification on Networks

Causal inference is the estimation of causal quantities (e.g., i's potential outcome $Y_i|do(t_i = t, z_i = z)$). However, only the statistical quantities (e.g., i's observed outcome $Y^i_{t_i, z_i}$) are available in observational data. To ensure that these statistical quantities can be used to infer the potential outcome (a.k.a., the causal identification problem), we make the following essential assumptions.

Assumptions. We make two lines of assumptions on networked data. First, we adapt the standard assumptions on independent data to the network setting following [7]:

Assumption1: Positivity. The probability of a unit with their neighbors to receive treatment or not is always positive, i.e., $\forall x, 0 < p(t_i = 1|x_i, \{x_j\}_{j \in \mathcal{N}_i}) < 1$.

Assumption2: Consistency. The potential outcome is same as the observed outcome under the same treatment assignment and peer exposure to neighbors , i.e., $Y_i|do(t_i = t, z_i = z) = Y^i_{t,z}$.

Assumption3: Strong Ignorability. Conditional to the features x_i and neighbors' features $\{x_j\}_{j \in \mathcal{N}_i}$, potential outcome $Y_i|do(t_i = t, z_i = z)$ is independent of treatment t_i and peer exposure z_i , i.e., $Y_i|do(t_i = t, z_i = z) \perp\!\!\!\perp t_i, z_i|x_i, \{x_j\}_{j \in \mathcal{N}_i}$.

In networked data, the standard SUTVA does not hold because of the presence of interference. Therefore, to identify the causal effects, we further assume the interference has the Markov property (i.e., *1-hop*) following [2] (here we set $T_{N_i} = \{t_j\}_{j \in \mathcal{N}_i}$ and $T_{-N_i} = \{t_j\}_{j \in -\mathcal{N}_i}$ for simplicity):

Assumption4: Markov. The potential outcome of a unit is only affected by their own and the immediate neighbors' treatments, i.e., $\forall\ T_{N_i}, T'_{N_i}, T_{-N_i}, T'_{-N_i}$ such that $Z(T_{N_i}) = Z(T'_{N_i})$, we have $Y_i|do(t_i = t, T_{N_i}, T_{-N_i}) = Y_i|do(t_i = t, T'_{N_i}, T'_{-N_i})$.

Identification. Given these assumptions, unit i's causal effects $\tau(x_i)$ (Eq. (2)) is identifiable. To avoid mess, we omit the subscription and denote by $x = (x_i, \{x_j\}_{j \in \mathcal{N}_i})$ in the following proof:

Proof.

$$\begin{aligned}
\tau(x) &= \mathbb{E}\left[Y|do(t = t', z = z') - Y|do(t = t'', z = z'')\middle|x\right] \\
&= \mathbb{E}\left[Y|do(t = t', z = z'))\middle|x\right] - \mathbb{E}\left[Y|do(t = t'', z = z'')\middle|x\right] \\
&= \mathbb{E}\left[Y|do(t = t', z = z')\middle|t = t', z = z', x\right] \\
&\quad - \mathbb{E}\left[Y|do(t = t'', z = z'')\middle|t = t'', z = z'', x\right] \qquad (3) \\
&= \mathbb{E}\left[Y_{t',z'}\middle|t = t', z = z', x\right] - \mathbb{E}\left[Y_{t'',z''}\middle|t = t'', z = z'', x\right]. \qquad (4)
\end{aligned}$$

Eq. (3) holds because of the "Strong Ignorability" assumption that given $x = (x_i, \{x_j\}_{j\in\mathcal{N}_i})$, the potential outcome $Y_i|do(t_i = t, z_i = z)$ is independent from the treatment t_i and peer exposure z_i. Eq. (4) is true because of the "Consistency" assumption. □

3 METHODOLOGY

In this section, we introduce our proposed method. We first prove why standard graph machine learning can not estimate causal effects. Then we breakdown the modules of NetEst in details.

3.1 Why Standard Graph Machine Learning Fails in Causal Inference?

We show that the failure of standard graph machine learning in estimating causal effects is due to two distribution mismatches between their objective functions.

We first introduce several functions with their corresponding notations. Following [37], we define a one-to-one projection function $\phi : \mathcal{X} \times 2^{\mathcal{X}} \to \mathcal{S}$, which maps a unit's own features and the neighbors' features into representation space $\mathcal{S}$. We denote $s_i = \phi(x_i, \{x_j\}_{j\in\mathcal{N}_i})$ as unit i's representation induced by ϕ. We will introduce the motivation of using this representation projection function in Sec. 3.2. We further define an estimation function $m : \mathcal{S} \times \{0, 1\} \times [0, 1] \to \mathcal{Y}$ that estimates the potential outcome from feature representation s_i, treatment t_i and peer exposure z_i. For simplicity, we also use a function $f : \mathcal{X}\times 2^{\mathcal{X}}\times\{0, 1\}\times[0, 1] \to \mathcal{Y}$ such that $f(x_i, t_i, z_i) = m(s_i, t_i, z_i) = m(\phi(x_i, \{x_j\}_{j\in\mathcal{N}_i}), t_i, z_i)$ to denote the whole estimation function starting from the original features. An estimation objective function needs a loss function, and we use the square loss in this paper. Now we can compare the objective functions of standard graph machine learning J_{ml} and causal effects estimation on networks J_{ce}. For simplicity, we remove the subscriptions in the following.

Objective function of machine learning J_{ml}. Standard graph machine learning estimates the potential outcome by optimizing the estimation loss over the networked observational data. Given network G, estimation function f, features x, treatment t, peer exposure z and outcome y, as stated in Sec. 2.2, the peer exposure z sufficiently represents the network effects induced by network G. Therefore, we can denote the observational data by the joint probability $p(x, t, z, y)$. Then the objective function of standard graph machine learning J_{ml} is:

$$J_{ml} = \int_{\mathcal{X}}\int_{\mathcal{T}}\int_{\mathcal{Z}}\int_{\mathcal{Y}} (f(x,t,z)-y)^2 p(x,t,z,y)\,dx\,dt\,dz\,dy \tag{5}$$

$$= \int_{\mathcal{X}}\int_{\mathcal{T}}\int_{\mathcal{Z}}\int_{\mathcal{Y}} (f(x,t,z)-y)^2 p(x)p(t|x)p(z|x,t)p(y|x,t,z)\,dx\,dt\,dz\,dy. \tag{6}$$

Note that the effects of network is encoded in the graph function f and peer exposure z, so G is not explicitly shown in Eq. (5). Eq. (6) is a chain rule expansion of Eq. (5). We can build a graph machine learning model f (e.g., GNNs) to predict the outcome by optimizing Eq. (6) on observational data.

Objective function of causal effects estimation J_{ce}**.** Causal inference is to estimate the causal effects $\tau(x)$ defined in Eq. (2) on network G. Therefore, given estimation model f, feature x, treatment t, peer exposure z and outcome y, the objective function of causal effects estimation J_{ce} is the estimation error of causal effects $\tau(x)$ over all units:

$$J_{ce} = \int_{\mathcal{X}} (\hat{\tau}(x) - \tau(x))^2 p(x)\,dx \tag{7}$$

$$\leq 8\int_{\mathcal{X}}\int_{\mathcal{T}}\int_{\mathcal{Z}}\int_{\mathcal{Y}} (f(x,t,z)-y)^2 p(x)p(y|t,x,z)\,dx\,dt\,dz\,dy, \tag{8}$$

where $\hat{\tau}(x)$ is the estimated causal effects. Eq. (8) is an upper bound for the objective function J_{ce}. Because directly optimizing the original objective function J_{ce} (Eq. (7)) is difficult, this upper bound can be used as an approximated objective function and the estimation function f can be built by optimizing it on the networked observational data. We use a general format of causal effect $\tau(x) = \mathbb{E}\left[Y_{1,z'}\middle|t = 1, z = z', x\right] - \mathbb{E}\left[Y_{0,0}\middle|t = 0, z = 0, x\right]$ following [7], which can be further decomposed as:

$$\tau(x) = \mathbb{E}\left[Y_{1,z'}\middle|t = 1, z = z', x\right] - \mathbb{E}\left[Y_{0,0}\middle|t = 0, z = 0, x\right] \tag{9}$$

$$= \mathbb{E}\left[Y_{1,z'}\middle|t = 1, z = z', x\right] - \mathbb{E}\left[Y_{0,z'}\middle|t = 0, z = z', x\right] \tag{10}$$

$$+ \mathbb{E}\left[Y_{0,z'}\middle|t = 0, z = z', x\right] - \mathbb{E}\left[Y_{0,0}\middle|t = 0, z = 0, x\right]. \tag{11}$$

Eq. (10) captures the individual effects of treatment and Eq. (11) models the peer effects from network interference. If we set $z = 1$, Eq. (9) becomes to the total effects. Therefore, Eq. (9) is a general format that contains all causal effects of interest in Sec. 2.2.

We then show the proof of upper bound in Eq. (8) as follows:

Proof. Given network G, model f, feature x, treatment t and outcome y, an empirical estimate can be denoted as $\hat{\tau}(x) = f(x, t = 1, z) - f(x, t = 0, 0)$, which can be similarly decomposed as $\hat{\tau}(x) = f(x, t = 1, z) - f(x, t = 0, z) + f(x, t = 0, z) - f(x, t = 0, 0)$. Finally the objective function of causal effects estimation J_{ce} [1] is as follows:

$$\begin{aligned}
J_{ce} &= \int_{\mathcal{X}} (\hat{\tau}(x) - \tau(x))^2 p(x)\,dx \\
&= \int_{\mathcal{X}} \big[f(x, t = 1, z) - f(x, t = 0, z) \\
&\quad - \big(\mathbb{E}(Y_{1,z}|t = 1, z, x) - \mathbb{E}(Y_{0,z}|t = 0, z, x)\big) \\
&\quad + f(x, t = 0, z) - f(x, t = 0, 0) \\
&\quad - \big(\mathbb{E}(Y_{1,z}|t = 0, z, x) - \mathbb{E}(Y_{0,z}|t = 0, 0, x)\big)\big]^2 p(x)\,dx \qquad (12) \\
&\leq 2\int_{\mathcal{X}} \big[f(x, t = 1, z) - f(x, t = 0, z) \\
&\quad - \big(\mathbb{E}(Y_{1,z}|t = 1, z, x) - \mathbb{E}(Y_{0,z}|t = 0, z, x)\big)\big]^2 p(x)\,dx \qquad (13) \\
&\quad + 2\int_{\mathcal{X}} \big[f(x, t = 0, z) - f(x, t = 0, 0) \\
&\quad - \big(\mathbb{E}(Y_{0,z}|t = 0, z, x) - \mathbb{E}(Y_{0,0}|t = 0, 0, x)\big)\big]^2 p(x)\,dx, \qquad (14)
\end{aligned}$$

where Eq. (12) is immediate with the definition of $\tau(x)$ and $\hat{\tau}(x)$. Eq. (13) and Eq. (14) are the estimation error of individual effects and peer effects, respectively. The inequality holds because $(a + b)^2 \leq 2(a^2 + b^2)$. For clearness, we conduct the proof of them separately.

We first focus on the individual effects Eq. (13):

$$\begin{aligned}
Eq.\ (13) = 2\int_{\mathcal{X}} &\big[f(x, t = 1, z) - \mathbb{E}(Y_{1,z}|t = 1, z, x) \\
&+ \big(\mathbb{E}(Y_{0,z}|t = 0, z, x) - f(x, t = 0, z)\big)\big]^2 p(x)\,dx
\end{aligned}$$

[1] With a square loss, J_{ce} is also know as Precision in Estimation of Heterogeneous Effects (PEHE) [16]

$$\leq 4\int_{\mathcal{X}}\left[f(x,t=1,z)-\mathbb{E}(Y_{1,z}|t=1,z,x)\right]^2 p(x)\,dx + 4\int_{\mathcal{X}}\left[f(x,t=0,z)-\mathbb{E}(Y_{0,z}|t=0,z,x)\right]^2 p(x)\,dx \quad (15)$$

$$\leq 4\int_{\mathcal{X}}\mathbb{E}\left[\left(f(x,t=1,z)-(Y_{1,z}|t=1,z,x)\right)^2\right]p(x)\,dx + 4\int_{\mathcal{X}}\mathbb{E}\left[\left(f(x,t=0,z)-(Y_{0,z}|t=0,z,x)\right)^2\right]p(x)\,dx \quad (16)$$

$$= 4\int_{\mathcal{X}}\int_{\mathcal{Y}}\left(f(x,t=1,z)-y\right)^2 p(x)p(y|t=1,x,z)\,dx\,dy + 4\int_{\mathcal{X}}\int_{\mathcal{Y}}\left(f(x,t=0,z)-y\right)^2 p(x)p(y|t=0,x,z)\,dx\,dy = 4\int_{\mathcal{X}}\int_{\mathcal{Y}}\sum_{t\in\{0,1\}}\left(f(x,t,z)-y\right)^2 p(x)p(y|t,x,z)\,dx\,dy \quad (17)$$

$$\leq 4\int_{\mathcal{X}}\int_{\mathcal{T}}\int_{\mathcal{Z}}\int_{\mathcal{Y}}\left(f(x,t,z)-y\right)^2 p(x)p(y|t,x,z)\,dx\,dt\,dz\,dy. \quad (18)$$

Eq. (16) can be obtained by Jensen's inequality. Given T is binary, we can unify Eq. (17) to Eq. (18) with integral. With a similarly technique, we have peer effects Eq. (14):

$$\begin{aligned} Eq.\ (14) &= 2\int_{\mathcal{X}}\Big[f(x,t=0,z)-f(x,t=0,0) \\ &\quad -\left(\mathbb{E}(Y_{0,z}|t=0,z,x)-\mathbb{E}(Y_{0,0}|t=0,0,x)\right)\Big]^2 p(x)\,dx \\ &= 2\int_{\mathcal{X}}\Big[f(x,t=0,z)-\mathbb{E}(Y_{0,z}|t=0,z,x) \\ &\quad +\left(\mathbb{E}(Y_{0,0}|t=0,0,x)-f(x,t=0,0)\right)\Big]^2 p(x)\,dx \\ &\leq 4\int_{\mathcal{X}}\int_{\mathcal{T}}\int_{\mathcal{Z}}\int_{\mathcal{Y}}\left(f(x,t,z)-y\right)^2 p(x)p(y|t,x,z)\,dx\,dt\,dz\,dy. \end{aligned} \quad (19)$$

Finally, add the upper bounds of individual effects Eq. (18) and peer effects Eq. (19), we can obtain an upper bound of the causal effects estimation error J_{ce} (Eq. (7)):

$$J_{ce} \leq 8\int_{\mathcal{X}}\int_{\mathcal{T}}\int_{\mathcal{Z}}\int_{\mathcal{Y}}\left(f(x,t,z)-y\right)^2 p(x)p(y|t,x,z)\,dx\,dt\,dz\,dy. \quad (20)$$

Eq. (20) is Eq. (8), concluding the proof. □

Distribution mismatch. Comparing Eq. (6) and Eq. (8), we find that standard graph machine learning actually models two more conditional probabilities $p(t|x)$ and $p(z|x,t)$ than the objective function of causal effects estimation. However, $p(t|x)$ and $p(z|x,t)$ are typically biased in observational data due to confounders (known as confounding bias [7, 37]). Consequently, a graph machine learning model trained on observational data will have biased estimations of the counterfactual outcomes and causal effects, because $p(t|x)$ and $p(z|x,t)$ are different in the counterfactual data. These distribution mismatches lead to the failure of applying standard graph machine learning models to estimate causal effects on networks.

How to fix it. To apply graph machine learning models for causal effects estimation on networks, the distribution gaps must be mitigated. By comparing Eq. (6) and Eq. (8), we find a sufficient (not necessary) solution is to force $p(t|x)$ and $p(z|x,t)$ as uniform distributions. In other words, causal effects estimation can be reduced into a multi-task graph machine learning problem on networked data. Namely, we can use a data-driven graph machine learning model, with some appropriate and sufficient losses that can force $p(t|x)$ and $p(z|x,t)$ to be uniformly distributed, to estimate the potential outcome. Although the original data generation can not be manipulated, we can achieve this goal by learning representations s_i for every unit i. Our model **NetEst** is motivated by these insights. For consistency, we still use $p(t|x)$ and $p(z|x,t)$ instead of s_i to denote the distributions to be uniformed throughout this paper.

Note that if the treatments are randomly assigned to units in a data collection, e.g., randomized controlled trials, these two conditional distribution $p(t|x)$ and $p(z|x,t)$ actually follow uniform distributions. In this case, it is safe to use a standard machine learning model to estimate causal effects. $p(t|x)$ is usually called propensity score in existing literature [34]. Similarly, we refer to $p(z|x,t)$ as the *peer exposure score* in this paper.

3.2 NetEst

Our model NetEst follows multi-task paradigm that uses graph machine learning to estimate causal effects on networks. NetEst is composed of four modules: Encoder, $p(t|x)$ Regularizer, $p(z|x,t)$ Regularizer and Estimator. Fig. 2 shows the overview of NetEst.

Encoder. The bias of propensity score $p(t|x)$ and peer exposure score $p(z|x,t)$ in observational data is caused by confounders. Traditional methods like matching [17, 36] can partially alleviate this by augmenting counterfactual data examples according to propensity score. However, we argue that a single scalar propensity score is not enough to capture the high dimensional confounders, especially on networked data. To capture both confounders from individual features x_i and neighbors' features $\{x_j\}_{j\in\mathcal{N}_i}$ while be flexible to later distribution regularization, we propose to learn representation for every unit on networks. In addition, because only immediate neighbors are assumed to have influences on a unit (Fig. 1), we just need to capture the features of i's 1-hop neighbors as $\{x_j\}_{j\in\mathcal{N}_i}$. Given this, we use Graph Convolutional Network (GCN) [23] as the representation function ϕ. A GCN layer aggregates the features of immediate neighbors according to a weight w.r.t. both the a unit's and his/her neighbor's degrees. The aggregated new features is then transformed to low-dimensional embeddings. Formally, given a network G, let $r_i^{(l)} \in \mathcal{R}^{d(l)}$ be the embedding of i in the l-th layer, where $d(l)$ is the embedding dimension of the l layer, the embeddings will be forwarded as:

$$r_i^{(l+1)} = \sigma\Big(\sum_{j\in\mathcal{N}_i}\frac{1}{\sqrt{d_i d_j}} r_j^{(l)} W^{(l)}\Big), \quad (21)$$

where $\sigma(\cdot)$ is a non-linear function, d_i and d_j are the degrees of units i and j, respectively. $W^{(l)}$ is a weight matrix of l-th layer, and $\mathcal{N}_i$ is the neighbors of node i. Note that because we need to retain the features of i, $\mathcal{N}_i$ also includes node i. Another benefit of using GCN is that it is applicable to both "inductive" and "transductive" settings, i.e., GCN could make predictions for a new network, or nodes within

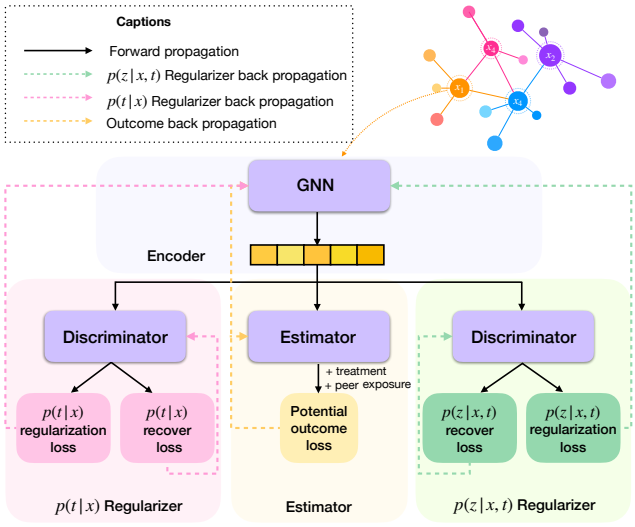

Figure 2: The overall framework of NetEst. NetEst is trained adversarially. The unit features and network structure are first encoded into embeddings via GNN. Then, the two discriminators in $p(t|x)$ regularizer and $p(z|x,t)$ regularizer are trained to recover treatment t and peer exposure z from embeddings by optimizing the $p(t|x)$ recover loss and $p(z|x,t)$ recover loss, respectively. With fixed parameters, the two well-trained discriminators optimize the encoder by the $p(t|x)$ regularization loss and the $p(z|x,t)$ regularization loss, together with the potential outcome loss given by the estimator. Solid lines are tensor forward propagation and dotted lines are loss back propagation. Note that the $p(t|x)$ regularization loss and $p(z|x,t)$ regularization loss are not used for the two discriminators although propagated through them.

the same network. This property enables us to estimate both the out-of-sample and within-sample causal effects. The final GCN layer produces the embeddings $s_i = \phi(x_i, \{x_j\}_{j\in\mathcal{N}_i})$, which encodes confounders from both a unit's own and neighbors' features.

p(t|x) Regularizer. Based on the embeddings s_i, we can uniform the propensity score $p(t_i|x_i)$ for every unit i. We propose to use adversarial training paradigm [8] to achieve this goal. Specifically, we first train a model (i.e., discriminator) that can recover the treatment t_i for every unit i from the fixed embeddings s_i as much as accurately. Formally, let $d_t : \mathcal{S} \to \{0, 1\}$ be the discriminator, it is trained by the $p(t|x)$ *recover loss* $\mathcal{J}_{rt}$ as:

$$\mathcal{J}_{rt} = -\frac{1}{V}\sum_{i=1}^{V}\big(t_i \log d_t(s_i) + (1-t_i)\log(1-d_t(s_i))\big). \tag{22}$$

Having the well-trained discriminator, we fix it as a "referee" to update the embeddings such that $p(t_i|x_i)$ is close to a uniform distribution. Given that treatment t_i is binary, the probability mass function of a uniformly distributed $p(t_i|x_i)$ is $p(t_i = 0|x_i) = p(t_i = 1|x_i) = 0.5$. Therefore, the $p(t|x)$ *regularization loss* $\mathcal{J}_{ut}$ used to uniform $p(t_i|x_i)$ is as:

$$\mathcal{J}_{ut} = \frac{1}{V}\sum_{i=1}^{V}(d_t(s_i) - 0.5)^2. \tag{23}$$

After many interactions of the "adversaries" between the encoder ϕ and discriminator d_t, the embedding s_i can finally be updated such that the discriminator d_t can not identify whether every unit receives treatment or not (both have 0.5 probability), i.e., $p(t|x)$ is forced into a uniform distribution.

p(z|x,t) Regularizer. Similar to the $p(t|x)$ Regularizer, we use another adversarial training paradigm to make the peer exposure score $p(z_i|x_i, t_i)$ uniformed for every unit i. A new discriminator $d_z : \mathcal{S} \times \{0, 1\} \to [0, 1]$ is first trained to recover the peer exposure z_i given embeddings s_i and treatment t_i via the following $p(z|x,t)$ *recover loss* $\mathcal{J}_{rz}$:

$$\mathcal{J}_{rz} = \frac{1}{V}\sum_{i=1}^{V}(d_z(s_i, t_i) - z_i)^2. \tag{24}$$

Then, we fix the discriminator d_z to update embeddings s_i to force the peer exposure score $p(z_i|x_i, t_i)$ into uniform distribution. Recall that z_i is defined as ratio of treated neighbors of i, and therefore is a continuous variable between 0 and 1. To approximate a continuous uniform distribution over range $[0, 1]$, we propose to uniformly sample a different value c_i^s~$[0, 1]$ for every unit i in every training iteration s, that is to say, every i has a varying label in every iteration. In this case, the predicted $\hat{z}_i = d_z(s_i, t_i)$ can be compared with any value from $[0, 1]$ with equal probability for multiple times. Hence, the randomly generated labels can mimic a continuous uniform distribution. Formally, the $p(z|x,t)$ *regularization loss* J_{uz} at iteration s is:

$$\mathcal{J}_{uz} = \frac{1}{V}\sum_{i=1}^{V}(d_z(s_i, t_i) - c_i^s)^2. \tag{25}$$

Note that as shown in Fig. 2, the $p(t|x)$ regularization loss $\mathcal{J}_{ut}$ and $p(z|x,t)$ regularization loss $\mathcal{J}_{uz}$ are only used to optimize the encoder, though they propagate gradients to their discriminators. We parameterize the two discriminators d_t, d_z with neural networks.

Estimator. Another objective is to minimize the observed outcomes estimation errors. We simply use neural networks as the estimator, which takes embeddings s_i, treatment t_i and peer exposure z_i as inputs to estimate the potential outcomes. Formally, for the estimator $m : S \times \{0, 1\} \times [0, 1] \to \mathcal{Y}$, we have the *potential outcome loss.* $\mathcal{J}_m$:

$$\mathcal{J}_m = \frac{1}{V}\sum_{i=1}^{V}(m(s_i, t_i, z_i) - Y^i_{t_i,z_i})^2. \tag{26}$$

Optimization. Algorithm. 1 shows the overall optimization procedure. NetEst optimizes the embedding s_i adversarially: (1) it first well trains the discriminators d_t and d_z by minimizing the $p(t|x)$ recover loss $\mathcal{J}_{rt}$ and $p(z|x,t)$ recover loss $\mathcal{J}_{rz}$, (2) then updates the estimator m with $\mathcal{J}_m$ and optimizes the encoder with a multi-task objective $\mathcal{J}_m + \alpha\mathcal{J}_{ut} + \gamma\mathcal{J}_{uz}$, where α and γ are coefficients that control the strengths of $p(t|x)$ and $p(z|x,t)$ regularization.

4 EXPERIMENTS

In this section, we evaluate the effectiveness of NetEst. We first set up the experiments and then report the results compared to baseline models. We further study why and when NetEst works.

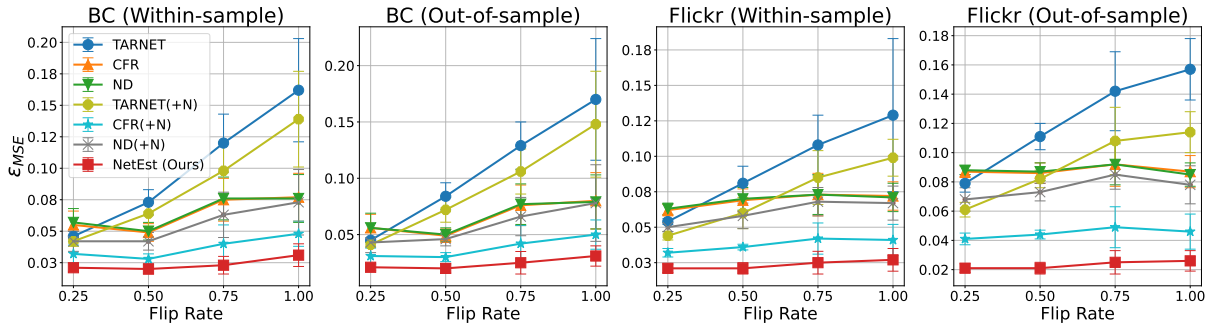

Figure 3: Counterfactual estimation errors ϵ_{MSE} v.s. percentages of units whose treatments are flipped (denoted as "flip rate"). From left: BlogCatalog "within-sample", BlogCatalog "out-of-sample", Flickr "within-sample", Flickr "out-of-sample".

Algorithm 1 The optimization of NetEstimator

Input: Network $G = \langle A, X \rangle$; the observed treatment t_i, peer exposure z_i and outcome $Y^i_{t_i,z_i}$; coefficients α and γ.
Output: Encoder ϕ, $p(t|x)$ Regularizer d_t, $p(z|x,t)$ Regularizer d_z and Estimator m.
Initialize ϕ, d_t, d_z and m;
for w = 1, 2, ..., W **do** ▷ Train model for W epochs
  for o = 1, 2, ..., O **do** ▷ Train d_t for O steps
    Compute $\mathcal{J}_{rt}$;
    Do one step of gradient descent for d_t:
    $\theta_{d_t}^{(o+1)} = \theta_{d_t}^{(o)} - \eta \nabla_{\theta_{d_t}} \mathcal{J}_{rt}$; ▷ η is learning rate
  for u = 1, 2, ..., U **do** ▷ Train d_z for U steps
    Compute $\mathcal{J}_{rz}$;
    Do one step of gradient descent for d_z:
    $\theta_{d_z}^{(u+1)} = \theta_{d_z}^{(u)} - \eta \nabla_{\theta_{d_z}} \mathcal{J}_{rz}$;
  for s = 1, 2, ..., S **do** ▷ Train ϕ and m for S steps
    Sample $c_i^s \sim [0,1]$ for every i;
    Compute $\mathcal{J}_m$, $\mathcal{J}_{ut}$, $\mathcal{J}_{uz}$;
    Do one step of gradient descent for ϕ and m:
    $\theta_{\phi}^{(s+1)} = \theta_{\phi}^{(s)} - \eta \nabla_{\theta_\phi} (\mathcal{J}_m + \alpha \mathcal{J}_{ut} + \gamma \mathcal{J}_{uz})$;
    $\theta_{m}^{(s+1)} = \theta_{m}^{(s)} - \eta \nabla_{\theta_m} \mathcal{J}_m$;
Return ϕ, d_t,d_z and m.

4.1 Experiments Setup

Datasets. For every unit i, only one treatment t_i, peer exposure z_i and outcome $Y^i_{t_i,z_i}$ can be observed(i.e., factual outcome). We can never know the groundtruth counterfactual outcome, and thus it is impossible to evaluate causal effects estimation directly. Therefore, following [14, 25, 39], we use semi-synthetic datasets, i.e., the networks (features, topology) are real but treatments and potential outcomes are simulated. We use two real-world social networks BlogCatalog and Flickr [14, 25]. In both datasets, a unit (node) is a user and an edge indicates their social relationship. Because the raw features of units are high-dimensional and very sparse, following [13, 25], we use LDA [5] to reduce the dimension to 10. "Out-of-sample" estimation requires we have a new network without observed outcomes, therefore, we use METIS [21] to partition the original network into three sub-networks as train/valid/test respectively. We evaluate the "within-sample" estimation on train networks and "out-of-sample" on the test network. Treatments and potential outcomes are simulated according to Fig. 1.

Treatments simulation. The treatment t_i is affected by i's features x_i and i's neighbors' features $\{x_j\}_{j \in \mathcal{N}_i}$. Let w_{X_1} be a randomly generated weight vector, then unit i's "propensity to treatment" pt_i is defined as $pt_i = \sigma(w_{X_1} \cdot x_i)$, where $\sigma(\cdot)$ is the sigmoid function. w_{X_1} mimics the causal mechanism of the confounders to treatments. We denote by pt_{N_i} the average of all i's neighbors' propensities, and denote by $tpt_i = \beta_x * pt_i + \beta_n * pt_{N_i}$ the total propensity to treatment of i. Then the treatment t_i is generated following:

$$t_i = \begin{cases} 1 & \text{if } tpt_i > \overline{tpt} \\ 0 & \text{else} \end{cases}, \tag{27}$$

where $\overline{tpt}$ is the average of all tpt_i. We set both β_x and β_n as 1. Given t_i and the network topology $\mathcal{A}$, the peer exposure z_i—the ratio of treated neighbors of i—can then be easily calculated.

Potential outcomes simulation. The potential outcome $Y_i|do(t_i, z_i)$ of i is affected by four factors: i's treatment t_i, peer exposure z_i, i's features x_i and i's neighbors' features $\{x_j\}_{j \in \mathcal{N}_i}$. We define "propensity to outcome" $po_i = \sigma(w_{X_2} \cdot x_i)$, where w_{X_2} is randomly generated to represent the causal mechanism of features to potential outcomes. Similarly, We let po_{N_i} be the average of all i's neighbors' propensities. Then the potential outcome is simulated by:

$$Y_i|do(t_i, z_i) = \beta_t \cdot t_i + \beta_z \cdot z_i + \beta_p \cdot po_i + \beta_o \cdot po_{N_i} + \epsilon, \tag{28}$$

where ϵ is a noise term. The parameters β_t, β_z, β_p and β_o are strengths to potential outcome of treatment, peer exposure, features, features of neighbors, respectively. We set β_t, β_z, β_p as 1 and β_o as 0.5. following the intuition that a unit's own features should have stronger effects than their neighbors. We use fixed parameters across networks because the causal mechanism is invariant.

Metrics. We consider two metrics: Mean Squared Error ($\epsilon_{MSE} = \frac{1}{V}\sum_{i=1}^{V}(\hat{y}_i - y_i)^2$) for counterfactual estimation where $\hat{y}_i$ and y_i are the estimated and groundtruth potential outcomes, respectively, and ($\epsilon_{PEHE} = \sqrt{\frac{1}{V}\sum_{i=1}^{V}(\hat{\tau}(X) - \tau(X))^2}$) for causal effects estimation, where $\hat{\tau}(X)$ is the estimation and $\tau(X)$ is groundtruth. Lower is better for both metrics.

Baselines. NetEst is compared with six baselines and three variants. **CFR** [37]: State-of-the-art model for causal effects estimation on independent data, which is optimized by the estimating

Table 2: Results of causal effects estimation. The PEHE error ϵ_{PEHE} (precision of estimating heterogeneous effects) is reported. The best is boldface while the second best is underlined. "N/A" means the model is not applicable for the peer effects.

Data (Setting)	effects	TARNET	CFR	ND	TARNET(+N)	CFR(+N)	ND(+N)	NetEst_U	NetEst_I	NetEst_P	NetEst
BC (Within-sample)	Individual	0.1140±0.0455	0.1292±0.0931	0.1442±0.0942	0.1315±0.0411	0.1121±0.0546	**0.0969±0.0422**	0.1207±0.0345	0.1088±0.0452	0.1139±0.0437	0.1186±0.0542
	Peer	N/A	N/A	N/A	0.4850±0.0104	0.3346±0.0439	0.4680±0.0321	0.1245±0.0491	**0.0632±0.0188**	0.0685±0.0176	0.0647±0.0188
	Total	0.9952±0.0811	0.8708±0.0931	0.8558±0.0941	0.9027±0.0852	0.5566±0.1373	0.7472±0.1135	0.4101±0.0358	0.2268±0.0970	0.2483±0.0791	**0.2214±0.1000**
BC (Out-of-sample)	Individual	0.1169±0.0457	0.1292±0.0931	0.1444±0.0944	0.1303±0.0406	0.1142±0.0540	**0.1014±0.0443**	0.1199±0.0399	0.1040±0.0457	0.1114±0.0469	0.1159±0.0516
	Peer	N/A	N/A	N/A	0.4830±0.0110	0.3347±0.0440	0.4682±0.0323	0.1243±0.0498	0.0630±0.0198	0.0679±0.0182	**0.0610±0.0181**
	Total	0.9903±0.0866	0.8707±0.0931	0.8557±0.0943	0.8952±0.0892	0.5555±0.1360	0.0.7438±0.1145	0.4051±0.0422	0.2205±0.1017	0.2436±0.0823	**0.2166±0.1043**
Flickr (Within-sample)	Individual	0.1029±0.0231	0.0760±0.0445	0.0926±0.0470	0.1195±0.0434	**0.0613±0.0306**	0.1483±0.0678	0.1855±0.0556	0.1529±0.0588	0.1632±0.0585	0.1513±0.0637
	Peer	N/A	N/A	N/A	0.4327±0.0177	0.2967±0.0370	0.4977±0.0066	0.0911±0.0188	**0.0612±0.0298**	0.0759±0.0184	0.0734±0.0284
	Total	1.0144±0.0620	0.9470±0.0704	0.9317±0.0711	0.8661±0.0701	0.5212±0.0593	0.8331±0.0755	0.3715±0.0634	**0.2996±0.0583**	0.3107±0.0669	0.3139±0.0545
Flickr (Out-of-sample)	Individual	0.1111±0.0215	0.0760±0.0445	0.0938±0.0467	0.1129±0.0376	**0.0604±0.0297**	0.1346±0.0568	0.1769±0.0543	0.1464±0.0592	0.1546±0.0627	0.1392±0.0646
	Peer	N/A	N/A	N/A	0.4220±0.0204	0.2967±.0370	0.4876±0.0134	0.0827±0.0209	**0.0544±0.0257**	0.0662±0.0141	0.0568±0.0243
	Total	0.9895±0.0648	0.9470±0.0704	0.9253±0.0621	0.8243±0.0654	0.5220±0.0586	0.7887±0.0822	0.3435±0.0647	0.2783±0.0572	0.2826±0.0586	**0.2732±0.0571**

observed outcomes estimation, and a so-called Integral Probability Metrics(IPM) that forces treated and control group to be closer. We use the Wasserstein distance implementation of IPM. **TARNet** [37]: a variant of CFR without IPM. **NetDeconf** [14]: extension of CFR to networked data, which uses GNN for encoding confounders, and Wasserstein distance for representations balancing. **CFR+(N)**, **TARNet+(N)**, **NetDeconf+(N)**: because the above three models do not consider interference, we add the peer exposure (+N) as extra input to them to evaluate their ability under network interference. **NetEst_U, NetEst_I, NetEst_P**: variants of NetEst without any regularizers ($\alpha = \gamma = 0$), only with $p(t|x)$ regularizer ($\alpha = 0.5, \gamma = 0$), and only with $p(z|x,t)$ regularizer ($\alpha = 0, \gamma = 0.5$), respectively.

Implementation details. We build our model as follow. We use 1 graph convolution layer as encoder[2]. We use 3 fully-connected layers for estimator and the two discriminators. All hidden embedding size is 32. Coefficient α and γ are set as 0.5. For hyperparameters, we use full-batch training and set the learning rate to 0.001 for all modules. All parameters are randomly initialized and updated by the Adam optimizer [22]. We run every task for five times (including simulation) and reported the average and 1-standard deviation. The experiment environment is an AWS *g4dn.4xlarge* instance.

4.2 Results Comparison

As stated in Sec. 2.2, we estimate the counterfactual outcomes and predict three interesting causal effects: individual effects, peer effects and total effects. For counterfactual estimation, a counterfactual treatments assignment T is over the entire network, we therefore simulate the counterfactual outcomes by flipping the treatments of randomly sampled subgroups of units. We try flip rates in $\{0.25, 0.5, 0.75, 1\}$ and report the counterfactual estimation errors in Fig. 3. In general, NetEst consistently outperform all baselines in "within-sample" and "out-of-sample" estimations on both datasets, suggesting the *effectiveness* of our model in handling the confounding bias. We notice all models' errors increase with a larger flip rate, but NetEst is still *robust*, showing a much lower error even we flip the treatments of 100% units. NetEst and its variants exhibit a similar superiority against baselines in predicting the three causal effects (Table. 2). We observe that NetEst works generally better than other models in estimating the total effects. This empirically demonstrates our conclusion in Sec. 3.1 forcing the $p(t|x)$ and $p(z|x,t)$ into uniform distributions is essential for causal effects estimation on networks, which is derived by studying the objective function of causal effects. We note that the two variants NetEst_U and NetEst_I works better under some settings. We speculate that forcing $p(t|x)$ into uniform distribution will also make $p(z|x,t)$ close to be uniformed and vice versa, since both regularizers basically enforce the embeddings to be close with each other (validated later in Fig. 5).

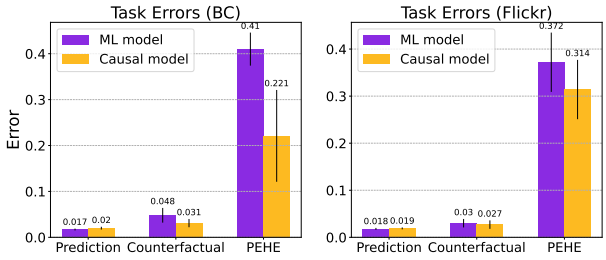

Figure 4: Errors of causal model NetEst and graph machine learning model (GCN) on three tasks: prediction, counterfactual estimation and causal effects estimation.

4.3 Why Does NetEst Work?

Motivation. We motivate NetEst by modifying the objective functions of graph machine learning models for causal effects estimation. To verify this modification, we compare NetEst with graph machine learning model GCN in Fig. 4. Causal model NetEst has worse performance on general prediction task compared to GCN but better on causal estimation tasks. It shows, as intended, forcing $p(t|x)$ and $p(z|x,t)$ into uniform distributions sacrifices the general prediction performance but alleviates the distribution mismatches and therefore favors beneficial for causal problem.

[2]Note 1 layer is consistent with the Markov assumption of network effects in Sec. 2.3. More layers may be necessary if network effects are beyond the 1-hop neighbors

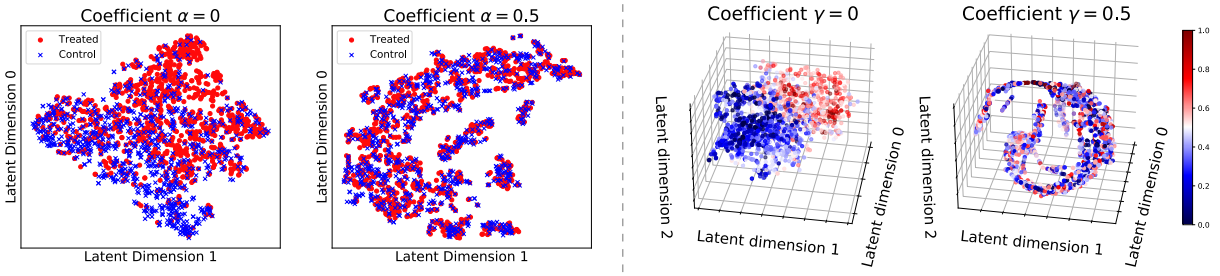

Figure 5: T-SNE projections of learned units' embeddings s_i. *Left*: without (α=0) and with (α=0.5) the regularizer for $p(t|x)$. A red point is a unit who was treated while a blue point means a controlled unit. *Right*: without (γ=0) and with (γ=0.5) the regularizer for $p(z|x,t)$. Units are colored by their observed peer exposures z. Red means a higher z, i.e., ratio of treated neighbors in this paper, while blue indicates a lower z. We use 3D projection for $p(z|x,t)$ as z is continuous.

Visualized interpretation. The uniform regularizers are conducted on the embeddings. We further visualize the learned embeddings s_i to understand why adversarial training works. We use t-SNE [38] to project units' embeddings s_i into 2-dimension colored by their binary treatments t_i and 3-dimension colored by their peer exposures z_i in for clearness in Fig. 5. With the distribution regularizers, the units points are highly overlapped on both figures. This overlapping means that for a given unit s_i, the discriminators d_t, d_z can not recover the treatment t and peer exposure z, suggesting t and z are uniformly distributed given s_i.

4.4 When Does NetEst Work?

The potential outcome scale varies a lot in observational data. We stratify units by their potential outcomes and break down the counterfactual estimation errors in Table. 3. We find NetEst works much better on moderate samples than extreme ones. We speculate NetEst can not alleviate the weakness of machine learning models on extreme data just with the proposed distribution regularizers. Understanding and solving this challenge is an interesting future direction.

Table 3: Counterfactual estimation errors according to potential outcome percentile. MSE error ϵ_{MSE} is reported.

Potential outcome strata	BC	Flickr
0-10%	$0.3008_{\pm 0.1529}$	$0.2642_{\pm 0.0784}$
10%-50%	$0.1192_{\pm 0.0375}$	$0.0966_{\pm 0.0143}$
50%-90%	$0.0614_{\pm 0.0235}$	$0.0536_{\pm 0.0060}$
90%-100%	$0.2423_{\pm 0.1756}$	$0.4767_{\pm 0.1368}$

5 RELATED WORK

Causal inference on independent data. Traditional causal inference is on independent data, where the Stable Unit Treatment Values Assumption (SUTVA) [35] guarantees potential outcome is not affected by the treatments of others. To alleviate the confounding bias in observational data, many existing works mimic the random treatment assignment from the observational data. A predominate approach is matching, which finds a similar peer for every unit from the opposite group [36]. Propensity score [17], describing the treated probability given features, is usually used as the criterion for matching. Another method is inverse probability of treatment weighting (IPTW) [3], which balances the data by re-weighting units based on their propensity scores. Recent works introduce representation learning to causal inference. [19, 20, 24, 37, 43, 44] learn embeddings for each unit via neural networks. The learned embeddings could predict the potential outcomes and are forced to be balanced between treated and control groups. Different from these methods on independent data, our focus is on networked scenarios, which has many practical use cases.

Causal inference on Networked Data. Unlike independent data, the units on networks are implicitly correlated, which violates the fundamental *SUTVA* assumption. Units on networks tend to behave similarly with their close neighbors (i.e., homophily [28]). They also affect each other (i.e., interference [18]). The dependencies between units on networks provide more complicated confounders, challenging the causal effects estimation. To infer networked causal effects from observational data, many works extend the methods on independent data into networks. [2] extends the back-door adjustment [33] into networks according to the causal graph built on networks. [7] introduces a summary variable of neighbors' treatments. They then extend the propensity score into networks to infer treatment effects. [26] applies Hilbert-Schmidt Independence Criterion (HSIC) [10] on networks to infer the treatments effects under interference. Our work follows the networked causal inference settings in [2, 7], but proposes an alternative method that adapts graph machine learning models for the causal effects estimation from the perspective of aligning their objective functions.

6 CONCLUSION

This paper studies causal effects estimation on networked data. We theoretically show the objective function of standard graph machine learning has two distribution mismatches against causal effects estimation, motivating our model NetEst that mitigates the distribution gaps via representation learning. Future works could study finding out the optimal treatment strategy, such as vaccine distribution plan, on networks based on estimated causal effects.

ACKNOWLEDGMENTS

This work was partially supported by NSF III-1705169, NSF 1937599, NSF 2119643, Okawa Foundation Grant, Amazon Research Awards, Cisco research grant USA000EP280889, Picsart Gifts, and Snapchat Gifts.

REFERENCES

[1] Roy M Anderson and Robert M May. 1985. Vaccination and herd immunity to infectious diseases. *Nature* 318, 6044 (1985), 323–329.

[2] David Arbour, Dan Garant, and David Jensen. 2016. Inferring network effects from observational data. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 715–724.

[3] Peter C Austin and Elizabeth A Stuart. 2015. Moving towards best practice when using inverse probability of treatment weighting (IPTW) using the propensity score to estimate causal treatment effects in observational studies. *Statistics in medicine* 34, 28 (2015), 3661–3679.

[4] Brian G Barkley, Michael G Hudgens, John D Clemens, Mohammad Ali, and Michael E Emch. 2020. Causal inference from observational studies with clustered interference, with application to a cholera vaccine study. *The Annals of Applied Statistics* 14, 3 (2020), 1432–1448.

[5] David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *the Journal of machine Learning research* 3 (2003), 993–1022.

[6] Zhixuan Chu, Stephen L Rathbun, and Sheng Li. 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.

[7] Laura Forastiere, Edoardo M Airoldi, and Fabrizia Mealli. 2021. Identification and estimation of treatment and interference effects in observational studies on networks. *J. Amer. Statist. Assoc.* 116, 534 (2021), 901–918.

[8] Ian J Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial networks. *arXiv preprint arXiv:1406.2661* (2014).

[9] Sander Greenland, Judea Pearl, and James M Robins. 1999. Causal diagrams for epidemiologic research. *Epidemiology* (1999), 37–48.

[10] Arthur Gretton, Olivier Bousquet, Alex Smola, and Bernhard Schölkopf. 2005. Measuring statistical dependence with Hilbert-Schmidt norms. In *International conference on algorithmic learning theory*. Springer, 63–77.

[11] Huan Gui, Ya Xu, Anmol Bhasin, and Jiawei Han. 2015. Network a/b testing: From sampling to estimation. In *Proceedings of the 24th International Conference on World Wide Web*. 399–409.

[12] Ruocheng Guo, Jundong Li, Yichuan Li, K Selçuk Candan, Adrienne Raglin, and Huan Liu. 2020. IGNITE: A Minimax Game Toward Learning Individual Treatment Effects from Networked Observational Data.. In *IJCAI*. 4534–4540.

[13] Ruocheng Guo, Jundong Li, and Huan Liu. 2020. Counterfactual evaluation of treatment assignment functions with networked observational data. In *Proceedings of the 2020 SIAM International Conference on Data Mining*. SIAM, 271–279.

[14] Ruocheng Guo, Jundong Li, and Huan Liu. 2020. Learning individual causal effects from networked observational data. In *Proceedings of the 13th International Conference on Web Search and Data Mining*. 232–240.

[15] William L. Hamilton, Rex Ying, and Jure Leskovec. 2017. Inductive Representation Learning on Large Graphs. In *Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS'17)*. 1025–1035.

[16] Jennifer L Hill. 2011. Bayesian nonparametric modeling for causal inference. *Journal of Computational and Graphical Statistics* 20, 1 (2011), 217–240.

[17] Keisuke Hirano and Guido W Imbens. 2004. The propensity score with continuous treatments. *Applied Bayesian modeling and causal inference from incomplete-data perspectives* 226164 (2004), 73–84.

[18] Michael G Hudgens and M Elizabeth Halloran. 2008. Toward causal inference with interference. *J. Amer. Statist. Assoc.* 103, 482 (2008), 832–842.

[19] Fredrik Johansson, Uri Shalit, and David Sontag. 2016. Learning representations for counterfactual inference. In *International conference on machine learning*. 3020–3029.

[20] Fredrik D Johansson, Nathan Kallus, Uri Shalit, and David Sontag. 2018. Learning weighted representations for generalization across designs. *arXiv preprint arXiv:1802.08598* (2018).

[21] George Karypis and Vipin Kumar. 1998. A fast and high quality multilevel scheme for partitioning irregular graphs. *SIAM Journal on scientific Computing* 20, 1 (1998), 359–392.

[22] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

[23] Thomas N. Kipf and Max Welling. 2017. 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*.

[24] Jing Ma, Yushun Dong, Zheng Huang, Daniel Mietchen, and Jundong Li. 2022. Assessing the Causal Impact of COVID-19 Related Policies on Outbreak Dynamics: A Case Study in the US. In *Proceedings of the ACM Web Conference 2022*. 2678–2686.

[25] Jing Ma, Ruocheng Guo, Chen Chen, Aidong Zhang, and Jundong Li. 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.

[26] Yunpu Ma and Volker Tresp. 2021. Causal inference under networked interference and intervention policy enhancement. In *International Conference on Artificial Intelligence and Statistics*. PMLR, 3700–3708.

[27] Laura Matrajt, Julia Eaton, Tiffany Leung, and Elizabeth R. Brown. 2020. Vaccine optimization for COVID-19: Who to vaccinate first? *Science Advances* 7, 6 (2020). https://doi.org/10.1126/sciadv.abf1374

[28] Miller McPherson, Lynn Smith-Lovin, and James M Cook. 2001. Birds of a feather: Homophily in social networks. *Annual review of sociology* 27, 1 (2001), 415–444.

[29] Razieh Nabi, Joel Pfeiffer, Murat Ali Bayir, Denis Charles, and Emre Kıcıman. 2020. Causal Inference in the Presence of Interference in Sponsored Search Advertising. *Workshop on Causal Discovery and Causality-Inspired Machine Learning* (2020).

[30] Elizabeth L Ogburn, Ilya Shpitser, and Youjin Lee. 2020. Causal inference, social networks and chain graphs. *Journal of the Royal Statistical Society: Series A (Statistics in Society)* 183, 4 (2020), 1659–1676.

[31] Elizabeth L Ogburn, Oleg Sofrygin, Ivan Diaz, and Mark J Van der Laan. 2017. Causal inference for social network data. *arXiv preprint arXiv:1705.08527* (2017).

[32] Elizabeth L Ogburn and Tyler J VanderWeele. 2014. Causal diagrams for interference. *Statistical science* 29, 4 (2014), 559–578.

[33] Judea Pearl. 2009. *Causality*. Cambridge university press.

[34] Paul R Rosenbaum and Donald B Rubin. 1983. The central role of the propensity score in observational studies for causal effects. *Biometrika* 70, 1 (1983), 41–55.

[35] Donald B Rubin. 1980. Randomization analysis of experimental data: The Fisher randomization test comment. *Journal of the American statistical association* 75, 371 (1980), 591–593.

[36] Donald B Rubin. 2006. *Matched sampling for causal effects*. Cambridge University Press.

[37] Uri Shalit, Fredrik D Johansson, and David Sontag. 2017. Estimating individual treatment effect: generalization bounds and algorithms. In *International Conference on Machine Learning*. PMLR, 3076–3085.

[38] Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of machine learning research* 9, 11 (2008).

[39] Victor Veitch, Yixin Wang, and David Blei. 2019. Using embeddings to correct for unobserved confounding in networks. In *Advances in Neural Information Processing Systems*. 13792–13802.

[40] Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. [n.d.]. Graph Attention Networks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*.

[41] Felix Wu, Amauri Souza, Tianyi Zhang, Christopher Fifty, Tao Yu, and Kilian Weinberger. 2019. Simplifying graph convolutional networks. In *International conference on machine learning*. PMLR, 6861–6871.

[42] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. 2019. How Powerful are Graph Neural Networks?. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*.

[43] Liuyi Yao, Sheng Li, Yaliang Li, Mengdi Huai, Jing Gao, and Aidong Zhang. 2018. Representation learning for treatment effect estimation from observational data. *Advances in Neural Information Processing Systems* 31 (2018), 2633–2643.

[44] Jinsung Yoon, James Jordon, and Mihaela van der Schaar. 2018. GANITE: Estimation of individualized treatment effects using generative adversarial nets. In *International Conference on Learning Representations*.

[45] Yuan Yuan, Kristen Altenburger, and Farshad Kooti. 2021. Causal Network Motifs: Identifying Heterogeneous Spillover Effects in A/B Tests. In *Proceedings of the Web Conference 2021*. 3359–3370.