IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 8 July 2024, accepted 28 July 2024, date of publication 2 August 2024, date of current version 12 August 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3437665

METHODS

Graph Neural Networks for Individual Treatment Effect Estimation

ANDREI SIRAZITDINOV[1], MARCUS BUCHWALD[1,2,3,4], VINCENT HEUVELINE[2,3,4], AND JÜRGEN HESSER[1,3,4,5,6,7], (Member, IEEE)

[1]Mannheim Institute for Intelligent Systems in Medicine, 68167 Mannheim, Germany
[2]Engineering Mathematics and Computing Laboratory (EMCL), 69120 Heidelberg, Germany
[3]Interdisciplinary Center for Scientific Computing (IWR), 69120 Heidelberg, Germany
[4]Heidelberg Institute for Theoretical Studies (HITS), 69120 Heidelberg, Germany
[5]Medical School, Heidelberg University, 69117 Heidelberg, Germany
[6]Central Institute for Computer Engineering (ZITI), 69120 Heidelberg, Germany
[7]CZS Heidelberg Center for Model-Based AI, 69120 Heidelberg, Germany

Corresponding author: Andrei Sirazitdinov (andrei.sirazitdinov@medma.uni-heidelberg.de)

This work was supported in part by the Data Storage Service Scientific Data Storage (SDS@hd), through the Ministry of Science, Research, and the Arts Baden-Württemberg [Ministerium für Wissenschaft, Forschung und Kunst (MWK)] and German Research Foundation (DFG) under Grant INST 35/1314-1 FUGG and Grant INST 35/1503-1 FUGG; in part by the Federal Ministry of Education and Research through projects Lean medical data: the right data at the right time (LeMeDaRT) under Grant 01ZZ2105A; in part by PerPain project under Grant 01EC1904B; in part by the Carl Zeiss Foundation Heidelberg Center for Model-Based Artificial Intelligence (AI) under Grant P2021-02-001; in part by BrainMEP under Grant BW1_1276/04; and in part by Erstellung eines qualitätsgesicherten Trainings-, Validierungs- und Testdatensatzes hepatozelluläres Karzinom (Q-HCC) under Grant 01 KD 2214. For the publication fee, we acknowledge financial support by Heidelberg University.

ABSTRACT Individual treatment effect (ITE) estimation is an important task for personalized decision-making in clinical settings. However, the data used to train an ITE estimation model may be limited. In this case, we expect that information regarding causal connectivity within features can facilitate model training and thus lead to better predictions. In this study, we incorporated causal information about the connectivity within features represented by a Directed Acyclic Graph (DAG) into the problem of ITE estimation. For this purpose, we propose a novel method based on Graph Neural Networks (GNN). Our results show that the proposed approach performs comparably to the current state-of-the-art methods on existing datasets. Using an artificial dataset, we demonstrate the potential advantages of using real graphs responsible for the data generation process over empty graphs with no edges. These advantages are particularly evident for datasets with limited training sizes and correctly defined DAGs. These findings highlight the potential of GNNs in personalized medicine for improving the assessment of individual treatment effects.

INDEX TERMS Causal inference, individual treatment effect estimation, graph neural networks.

I. INTRODUCTION

Causal inference plays an important role in social, medical, and epidemiological sciences [1], [2]. Estimating an Individual Treatment Effect (ITE) is a crucial task that can lead to personalized and effective therapy selection. This can be performed using data from randomized control trials (RCTs) [3] or observational studies [4]. RCTs are necessary to properly assess causal effects; however, they can be expensive and unethical. For these reasons, real-world RCT experiments may be imperfect, and thus have potentially insufficient power owing to the small number of participants [5]. The nonlinear relationships between the covariates and outcomes may further complicate the analysis. To mitigate these drawbacks, causal inference methods can be used to estimate the effects from the RCT data. In observational studies, the data quality may be a limiting factor, forcing researchers to work with only a small fraction of the data. In both cases, the small amount of available information poses a significant challenge for ITE estimation algorithms.

The main motivation of this study is to propose a method that can use existing and validated knowledge regarding causal dependencies within covariates to improve ITE estimation based on observational data. However, this method can also be applied to RCT data. Our main hypothesis is that the use of structural information can reduce model

The associate editor coordinating the review of this manuscript and approving it for publication was Liangxiu Han.

complexity and thus improve outcome estimation when the size of the training set is limited. When the training set is not limited, we assume a performance similar to that of the other methods. We believe that this can be achieved by using information from causal graphs. Causal graphs are used to identify confounders (variables that influence both the treatment and outcomes), to find variables that must be included in the model for unbiased causal effect estimation, and, finally, to visualize causal dependencies within features. However, outside of the Structural Causal Models (SCM) framework [6], information about connectivity within features is often discarded, and outcomes are predicted solely on the covariates.

To make ITE estimation from observational studies unbiased and general, we rely on the following assumptions [7]: consistency, exchangeability, positivity, Stable Unit Treatment Value Assumption (SUTVA), and unconfoundness. Consistency [7] states that the potential outcome for a patient receiving the observed treatment coincides precisely with the observed outcome for that treatment. Potential outcomes are the outcomes that can be hypothetically obtained from different treatments. The second assumption, exchangeability, implies that the treatment is well-defined such that no variation is present when performing the treatment. Positivity means that every patient must have a non-zero probability of receiving each of the considered treatments [8]. SUTVA is a combination of the consistency assumption and the assumption of ''no interference,'' i.e., the potential outcomes of a patient are independent of the treatment assignment of other individuals [9]. Finally, in the case of ITE estimation based on observational data, the assumption of unconfoundness is made, which reflects that variables outside the observed ones influence neither the treatment assignment nor the outcome.

In this study, we presume that covariates contain causal relationships among themselves and causally influence both, treatment and outcome. A causal relationship between two variables means that a change in the value of the first variable changes the value of the second variable but not necessarily if the order is reversed. A Directed Acyclic Graph (DAG) represents this information. The directed edges define the cause-effect relationship between two features (vertices). As the treatment outcome depends more on some features than on others, information flows through the weighted graph edges and nodes.

Graph Neural Networks (GNNs) [10] have proven to be important tools for working with graph information [11]. This study shows how GNNs can be used for causal effect estimation, given that we have a dependency graph of covariates. We hypothesize that using GNNs for ITE estimation helps leveraging information about causal links between features, if available, while being more computationally efficient than other methods for tabular datasets of limited size.

To investigate our hypothesis, we create a deterministic artificial dataset accompanied by a dependency graph, in which the values of the nodes are equal to the sum of the values of their parents. We assume that ITE estimation on such a dataset is challenging for models that do not use information about connectivity within features. In contrast, models that consider the structure of a dataset as a graph would show advanced ITE estimation performance. We also create DAGs for existing observational datasets and test our model on them. We provide an overview of related approaches, describe our method, compare it with state-of-the-art techniques on artificial and existing datasets, discuss and summarize the results, and provide an outlook for future work.

II. RELATED WORK

We work within the potential outcomes framework [12], in which we assume actual, potential, and counterfactual treatment outcomes for a given subject. The actual outcomes are the observed outcomes of a given treatment. Potential outcomes are the set of all possible outcomes for each individual under each possible treatment. Counterfactuals are potential outcomes that have not been observed. Various methods for estimating the ITE using neural networks within this framework have been published. They can be divided into meta-learners [13], representation-based [14], adversarial [15], and variational [16]. For a detailed review, please refer to [17]. Meta-learners such as S-Learner and T-Learner [13], differ in the way they pass the covariates and treatment through the fully connected layers. In the case of S-Learner, the covariates and treatment are concatenated and passed jointly through the network. This potentially makes it more difficult for the networks to distinguish between input features and treatment in a high-dimensional space. Unlike S-learners, T-learners use different networks for each treatment, but this can lead to increased variance. In contrast to meta-learners, representation-based methods [14], [18] address the above shortcomings by first transforming the input dataset into a hidden space over multiple fully connected layers. With the exception of the Treatment Agnostic Representation Network (TARnet) [14], they further minimize the discrepancy between the distributions of treated and untreated subjects using, for example, Wasserstein integral probability metric [19] in case of the Counterfactual Regression with Wasserstein Integral Probability Metrics (CFR-Wass) [14]. The potential outcomes are then estimated over a number of treatment-specific branches. This results in less bias due to distributional shifts between treatment groups [14].

Adversarial methods such as Generative Adversarial Nets for inference of individualized Treatment Effects (GANITE) [15], which is based on Generative Adversarial Networks (GANs) [20], attempt to trick the discriminator network into believing that the generator is predicting real counterfactual outcomes. Variational methods such as Treatment Effect with Disentangled Autoencoder (TEDVAE) [16] use a DAG to model the relationship between the distributions of the treatment, outcome, and covariates. Unlike our method, which uses GNNs, namely spatial-based convolutional graph neural

networks [21], [22], variational methods do not consider information regarding relationships within covariates.

GNNs were utilized to solve various tasks for the connected subjects. Examples include traffic prediction [23], [24], scene graph generation [25], online shopping recommendations [26] and drug discovery [27]. For a comprehensive overview of the underlying techniques, please refer to [22] and [28]. Multiple endeavors have been undertaken to incorporate graph knowledge into classification and regression tasks using tabular data [29]. However, such methods often focus on the correlation between features rather than causality, except for of the method proposed by Zhai et al. [30], in which causality is included in the model in the context of online advertising.

Several attempts have been made to add structural information to infer the causal effects. Zečevič et al. [31] demonstrated that structural information in the form of a graph used by a GNN is related to structural causal models [6]. They demonstrated how to implement interventions in GNNs, similar to those in structural causal models. Ensuring that the statistical properties are reflected in the model helps to learn the desired data representation and improve model performance. Wein et al. [32] used a GNN to combine structural and functional information in a framework that can predict brain dynamics. In a similar study, Chu et al. [33] estimated the treatment effect from observational data of related subjects. Parafita and Vitria [34] used graphs for causal effect estimation; however, their method did not rely on convolutional graph neural networks and thus differs from our approach. We propose a model with no dependencies between subjects but with a structure, that is, a graph of the characteristics of a subject that defines the data-generation process. The contributions of this study are as follows:

1) Introduction and description of a novel method that, in contrast to other state-of-the-art methods, can use graph information for ITE estimation.
2) Evaluation and comparison of the proposed method with state-of-the-art ITE estimation methods on existing and artificial datasets.

III. METHOD

A. PROBLEM FORMULATION

Suppose we are given a dataset from observational studies $D = \{[x_i, y_i, t_i]\}_{i=1}^{N}$, where $x_i \in X \in \mathbb{R}^M$ is a set of covariates, the treatment $t_i \in T \in \{0, 1\}$ is binary, and the outcome vector $y_i \in Y$ can be discrete or continuous. We also assume that we have a weighted DAG $G = (V, E)$, describing the data generation process of dataset D, with vertices $V \in \mathbb{R}^M$ and edges $E \in \mathbb{R}^K$. The graph can be represented by an adjacency matrix $A \in \mathbb{R}^{M\times M}$ with weights $W \in \mathbb{R}^{M\times M}$. We expect the adjacency matrix A to be upper triangular matrix with elements having a value of w_{ij} if there is a directed edge between parent node v_i and child node v_j. We assume that all graph nodes are sorted such that $i < j$.

Our goal is to estimate the individual treatment effect or the Conditional Average Treatment Effect (CATE), defined as

$$\tau(x_i) = \mathbb{E}[Y^1 - Y^0 | X = x_i], \tag{1}$$

where Y^1, Y^0 are potential outcomes that would have been observed if the i-th subject from the dataset with covariate value x_i was assigned to the treatment or control group, respectively. The definition can be augmented by including adjacency matrix A in the model to account for connectivity within the data.

$$\tau(x_i) = \mathbb{E}[Y^1 - Y^0 | X = x_i, A]. \tag{2}$$

To evaluate the performance of a model, we compute the Precision in Estimating Heterogeneous Effect (PEHE) as

$$\epsilon_{\text{PEHE}} = \frac{1}{N}\sum_{i=0}^{N}(\tau(x_i) - \hat{\tau}(x_i))^2, \tag{3}$$

where $\hat{\tau}(x_i)$ represents the result of the ITE estimation algorithm. We also compute the Average Treatment Effect for the entire population as

$$\epsilon_{\text{ATE}} = |\frac{1}{N}\sum_{i=0}^{N}(\tau(\boldsymbol{x}_i) - \hat{\tau}(\boldsymbol{x}_i))|. \tag{4}$$

To calculate policy risk, which measures the expected loss if treatment is carried out according to the ITE policy prescribed by the algorithm, we use the following formula [35]:

$$\begin{aligned}\mathcal{R}_{pol} = 1 - \mathbb{E}[Y^1|\pi(X) = 1]P(\pi(X) = 1)\\ + \mathbb{E}[Y^0|\pi(X) = 0]P(\pi(X) = 0),\end{aligned} \tag{5}$$

where $\pi(X) = 1$ if $Y^1_{RCT} - Y^0_{RCT} > 0$ and $\pi(X) = 0$ otherwise. The error of the average treatment effect on the treated can be calculated as [1]:

$$\epsilon_{\text{ATT}} = |\text{ATT} - \frac{1}{|T|}\sum_{i\in T}\tau(\boldsymbol{x}_i)|, \tag{6}$$

where ATT is the average treatment effect on the treated, computed as $\text{ATT} = |T|^{-1}\sum_{i\in T} y_i - |C \cap E|^{-1}\sum_{i\in C\cap E} y_i$, where C is a control group if all treated subjects T are from randomized sample E, and $|\cdot|$ defines the number of subjects in a set.

B. GRAPH NEURAL NETWORKS

GNNs [22] consist of several graph convolutional layers [36] that receive embeddings of node values, adjacency matrix A, and weight matrix W as the input. The GNNs update a state of the node using information from its neighbors. They can be used for both directed and undirected graphs. Because we are working with DAGs, we assume that a node can only be updated using the information from its parents.

The forward pass for node updates encompasses three main steps: message preparation, aggregation, and update [37]. The goal of message preparation is to refine the information from parents. This is performed by passing the embeddings of the

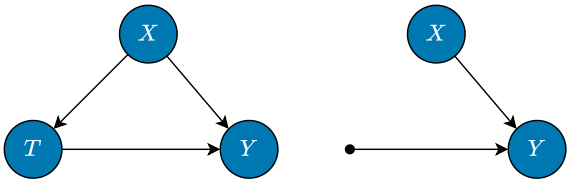

FIGURE 1. Causal graphs before (left) and after (right) intervention on node *T*.

parent node through a learnable function and multiplying the results by the edge weights W. The messages from the parents are then aggregated by summation, averaging, or choosing the maximum value. During the update step, aggregated messages are combined with node representations by concatenation, multiplication, addition, or passing through the gated recurrent unit [38], with subsequent passes through several fully connected layers [39], [40].

C. INTERVENTION ON GNN

Under the assumption that a new node value h_i depends on the values calculated in the previous step, Zečević et al. [31] stated that the update rule for a graph with intervention on the k-th node d_k involves removing all the edges incoming to this node. Specifically, in this case,

$$h_i = \phi\left(d_i, \bigoplus_{j \in \mathcal{M}_i} w_{ij}\psi(d_j)\right), \tag{7}$$

where $\mathcal{M}_i$ is the set of neighbors of node i, constructed in such a way that if there is an intervention on node d_k, it no longer includes information from its parents pa_i. Formally, $\mathcal{M}_i = \{j | j \in \mathcal{N}_i, j \notin \text{pa}_i \iff j = k\}$. In this context, pa_i denotes the parents of node i, $\mathcal{N}_i$ is the set of all neighbors of node i, ψ and ϕ are learnable functions, w_{ij} is the edge weight between nodes i and j, $\bigoplus$ represents an aggregation function such as summation, averaging, or taking the maximum value, and $\iff$ means if and only if.

This intervention corresponds to the $do(d_k)$ operator described by Pearl [6]. Considering the intervention on the treatment, we are left to find a graph for covariates and outcomes without including the treatment because we assume that the treatment variable always affects the outcome and that no other edges are coming to it because of the intervention. Fig. 1 shows the intervention at node T for a simple causal graph.

D. FINDING THE CAUSAL GRAPH

Acquiring the causal structure of a dataset can be achieved using existing causal discovery methods. The most popular are the Linear non-Gaussian Acyclic Model (LiNGAM) proposed by Shimizu et al. [41], the Peter-Clark (PC) algorithm [42], and Greedy Equivalence Search (GES) [43]. For an overview of these methods, please refer to Glymour et al. [44].

To find the adjacency matrix A of the graph using causal discovery methods, we combine the variables X and the

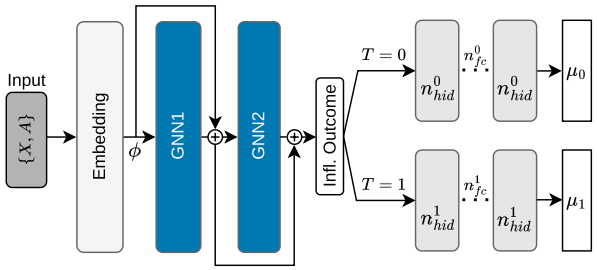

FIGURE 2. The GNN-TARnet architecture.

available outcomes Y into a dataset $\{X, Y\}$. Before running the causal discovery algorithm, we mark the node corresponding to outcome Y as the only node with no children, assuming that it has no influence on other variables. Then we run the causal discovery algorithm and extract a graph from the adjacency matrix. This is achieved by determining the coordinates of the non-zero elements. Finally, after removing all edges to the outcome node, we store the coordinates as an array of tuples, where the first element is the parent node and the second is a child node. The indices of the nodes influencing the outcome are stored separately. If the causal discovery method fails to identify any nodes that influence the outcome, we use an empty graph having an identity adjacency matrix. An empty graph with an identity adjacency matrix is referred to as the *identity* graph.

E. GNN-TARnet

With the above assumptions, we propose a method called GNN-TARnet (Graph Neural Network Treatment Agnostic Representation Network), which can take into account information about causal relationships between covariates to predict ITE. This method is similar in design to the TARnet proposed by Shalit et al. [14]. We based our method on the TARnet architecture because of its simplicity and a solid performance compared to other methods on the existing datasets [17]. We note that an architecture based on reducing the discrepancy between the distributions of the treated and untreated, such as CFR-Wass [14], was not chosen as the basis because we wanted to see the effect of using DAGs in the context of ITE estimation without overcomplicating the model. Our approach can be extended to include discrepancy reduction techniques, but this is beyond the scope of this paper.

The main difference of GNN-TARnet compared to the TARnet is the computation of the hidden representation before branching (Fig. 2). In our case, we use graph convolutional layers [36] instead of fully connected layers, followed by two treatment-specific branches. To train the network, we minimize the following loss function:

$$\mathcal{L} = \mathbb{E}[(1-T)(\mu_0(X,A) - Y)^2 + T(\mu_1(X,A) - Y)^2]. \tag{8}$$

The loss function $\mathcal{L}$ is equal to the expected value $\mathbb{E}$, which is the average of the squared differences between the observed outcome Y and the estimated outcomes μ_0 and

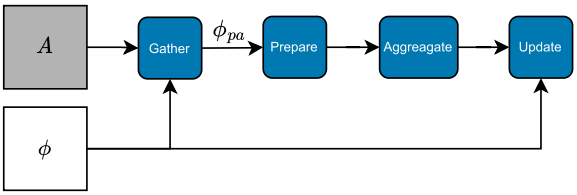

FIGURE 3. GNN block structure.

μ_1 under the control and treatment conditions weighted by the probabilities of not receiving treatment $(1 - T)$ and receiving treatment T.

The network receives the input covariates X and the adjacency matrix A. As information regarding the edge weights W is rarely available, we set their values to one. First, the covariates, which also act as graph node values, are passed through the embedding layers, where the original information is combined with the trained neural network weights and converted to the same dimension as the embedding dimension of the GNN layer. Each covariate has an embedding that is independent of the other covariates. This is achieved by reshaping the covariates before embedding to obtain one dimension more. Embeddings and edges are passed through a GNN block (Fig. 3). The GNN block extracts node indices and causal parents from the adjacency matrix. The parent representations ϕ_{pa} are then gathered from the node representations using the parent indices. The parent messages are prepared in the next step by passing the parent representations through fully connected layers without bias [37]. Messages from parents are then aggregated [37]. In our case, messages to the current node from the parent nodes are summed. Finally, the node representations are updated with aggregated messages from the parents by adding or multiplying them and passing them through fully connected layers [39]. We also used skip connections [45] after each GNN block, as we observed that adding them improves the performance.

After these steps, we obtain a vector for the updated node embeddings. If we know the adjacency matrix between X and Y, we condition the nodes directly affecting Y, flatten the hidden representations, and pass them through the treatment-specific branches. When only an identity graph is available, we impose an equally weighted influence of the nodes on the outcome variable and condition on all available nodes. Finally, the losses are calculated, and the entire system weights are updated using backpropagation.

IV. EXPERIMENTS

An intrinsic issue in validating the performance of models in the field of causal inference is that counterfactual outcomes are unavailable from real-world data. Hence, we evaluated the algorithm performance on publicly available datasets such as IHDP [14], [46], [47] and JOBS [48].

The IHDP dataset is from the Infant Health and Development Program [46] with artificial outcome settings "A" and "B." The settings differ in the manner in which the outcomes are generated. Setting "A" is linear, and "B" as exponential. We denote the corresponding datasets as IHDP_A and IHDP_B. From [47], it is known that for IHDP_A and IHDP_B only a few randomly selected covariates influence the outcome. It means that the other covariates have no influence to the outcome. Both the datasets include 25 covariates and 747 participants.

The JOBS dataset contains 3,212 cases defined by eight factors that characterize population and household income in 1974 and 1975. Individuals in the treatment group received specialized professional training. Employment status is a binary outcome.

We also compared the performance of GNN-TARnet with various graph construction methods: the identity graph (GNN-TARnet (ident.)), LiNGAM [41] (GNN-TARnet (LiNGAM)), PC [42] (GNN-TARnet (PC)), and GES [43] (GNN-TARnet (GES)) to TARnet and other methods such as meta-learners SLearner [13] and TLearner [13], CFR-Wass [14], TEDVAE [16], and GANITE [15] to determine the benefits and shortcomings of the proposed method on tabular datasets such as JOBS, IHDP_A, and IHDP_B.

It is difficult to evaluate the performance of our method because no datasets are currently available for ITE estimation accompanied by causal graphs. Thus, to understand the limitations and advantages of GNN-TARnet as well as to test the feasibility of the data model represented by DAG, we created an artificial dataset with a layered structure called SUMmation (SUM), with covariate values equal to the sum of other covariates and potential outcomes generated as a sum or average value of nodes influencing the outcome. The intention was to create a structured dataset mimicking real-world causal relationships, ensuring that the model could be evaluated under reliable and measured controlled conditions. The details are presented in the next section.

A. SUMMATION DATASET

The graph structure of the SUM dataset was inspired by the graphs presented in the *bnlearn* [49] repository of Bayesian networks. The graphs from the repository have varying numbers of layers, with nodes in one layer being the only parents to nodes in subsequent layers. This indicates that there are no intra-layer connections. The number of root nodes in the paths from the roots to a leaf node is often greater than or equal to the number of children of the root nodes. In addition, a layer before the leaf node (output nodes) usually contains significantly fewer nodes than the total number of nodes in the entire graph. The average degree for most of the graphs in the repository was two for small graphs and around three for the other networks. The maximum number of incoming edges to a node is equal to 13. To create the dataset, we first created a layered graph and then generated its node values. A detailed explanation of these steps is provided below.

1) CREATING LAYERED GRAPH

Suppose that we are given a number of layers l. To create the first layer of graph G, we generate a range of length $r + 1$,

which stores consecutive integer values between zero and r. Hence, $r+1$ defines the total number of root nodes. We then stored the array as a value in dictionary L, where the layer number represents the key. All subsequent dictionary entries store ranges of length k, starting with the values immediately following the last element of the array stored in the previous layer. For the dataset, length k is a random number chosen between three and eight, and if the layer is the final layer, then the number of nodes in it is $m = 0.3$ times the total number of nodes in the previous layers combined. The nodes in the subsequent layer are potential children of the parent nodes in the previous layer. The pseudocode for generating dictionary L that stores the graph layers is presented in Algorithm 1. To finally construct graph G, we created edges between individual parent nodes and k children nodes, where k is an uniformly selected from the range between zero and k random number. The nodes in the output layer are connected to all the nodes in the previous layer. Thus, we guarantee that every internal graph node contains a parent and child node.

Scalar m was chosen such that the number of output nodes was always greater than two, but much less than the total number of nodes in the previous layers. We chose r as a random number between ten and 20 because, in this case, the number of root nodes is always greater than the number of k nodes in the next layer. k is chosen such that the average node degree is in a range similar to that in the *bnlearn* repository. The choice of parameters k, r and m makes our graphs similar to those from the *bnlearn* repository.

Algorithm 1 Generate Layers

```
Given: number of layers l
m ← 0.3
Initialize an empty dictionary L
r ← random number between 10 and 20
L[0] ← range of length r + 1 from 0 to r
for i ← 1 to l do
    if i = l then
        k ← ⌊m × (∑_{j=0}^{l−1} length(L[j]))⌋
    else
        k ← random number between 3 and 8
    end if
    L[i] ← range of length k starting from the last element of L[i − 1] + 1
end for
```

2) GENERATING NODE VALUES

After creating the graph, we generated node values as follows. The values of the nodes in the first layer were uniformly randomly sampled between 0 and 1. The values of nodes in the next layer are equal to the sum of those of their parents. The node values in the last layer influence the outcome generation.

To make the data resemble those from observational studies, the treatment was assigned as follows. We found the average value of all outcome nodes for all subjects. For each subject, if the mean of the outcome nodes of an individual subject was greater than the mean of the outcome nodes for all subjects, the treatment was assigned a value of one; otherwise, it was assigned a value of zero.

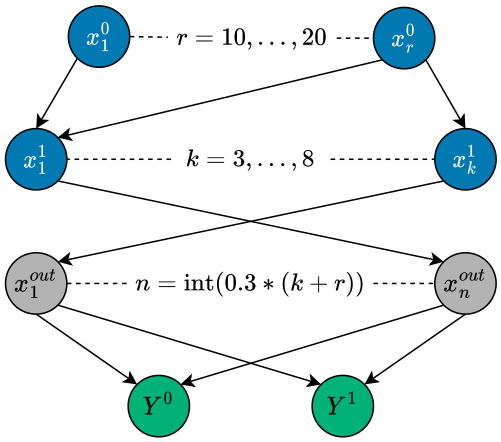

FIGURE 4. DAG for SUM dataset with two layers. Nodes in the output layer are marked as gray, potential outcomes as green, and the nodes of layers zero and one are blue. Not all edges and nodes are presented.

Mathematically speaking, for the k-th subject x_k in a data set with N subjects and set O of all node indices affecting the outcome with index o treatment t_k is assigned as:

$$t_k = \begin{cases} 1, & \text{if } \frac{1}{n}\sum_{j\in O} x_{k,j} > \frac{1}{n\cdot N}\sum_{i=1}^{N}\sum_{j\in O} x_{i,j} \\ 0, & \text{otherwise,} \end{cases} \tag{9}$$

where $O = \{j | j \in pa_o,\ j = 1 \ldots M\}$.

The outcomes were generated as the sum of the output node values in the case of treatment and mean values in the case of no treatment. Fig. 4 shows an example of a graph of the SUM dataset with two layers.

The SUM dataset was used to examine the importance of using a causal graph versus an identity graph. We ran the experiments with 16, 32, 64, and 128 subjects in the training set and up to four graph layers. These values were chosen because they allow us to demonstrate the model performance for different scenarios and complexities and because increasing the numbers does not change the overall model performance. The test set was fixed and comprised of 120 data points. In addition, we report the performance of TARnet on this dataset.

B. NODE MASKING

By design, GNN-TARnet relies on nodes that influence the outcomes to make predictions, where the nodes correspond to embedding values of input variables passed through GNN layers. Theoretically, this means that knowing all such nodes, if no other nodes influence them, can greatly simplify the training. We want to know how successful the causal discovery methods are in finding such nodes in the existing datasets and whether there is a structure behind them. To do

TABLE 1. Parameters of GNN-TARnet model.

	n_{fc}^0	n_{fc}^1	n_{hid}^0	n_{hid}^1	n_{fc}^{gnn}	n_{hid}^{gnn}	bs	lr
IHDP_A	7	6	96	240	4	128	64	1e-4
IHDP_B	4	6	208	240	4	128	32	1e-4
JOBS	5	6	240	176	2	16	64	1e-2
SUM	2	2	16	16	2	16	2	1e-3

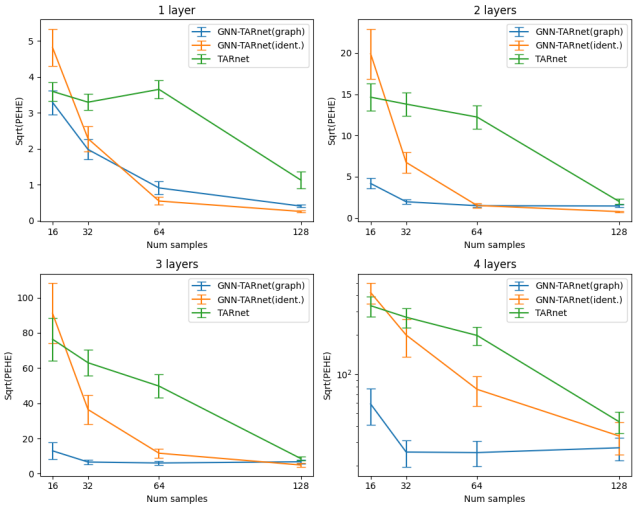

FIGURE 5. Dependency of number of layers and $\sqrt{\epsilon_{\text{PEHE}}}$ for different number of samples in the training set.

this, we replace or mask the values of the covariates identified as influencing the outcome with zeros. Our assumption is that if the causal discovery method correctly identifies all nodes influencing the outcome and there are no other nodes influencing the outcome, the prediction result will be the same as replacing all values in the dataset with zeros. We report the performance of GNN-TARnet (LiNGAM) on the datasets with all covariates masked with zeros and only covariates influencing the outcome replaced by zeros. In the SUM dataset, we mask all covariate values corresponding to the nodes of the last layer of graph G by replacing them with zeros while preserving the original graph structure. Note that masked output nodes are not counted as a layer, which is done to make inference on the dataset more difficult; otherwise, one can train a model on the variables that influence the outcome and discard all other variables to obtain the best possible predictions.

C. IMPLEMENTATION DETAILS

The number of layers and hidden units for treatment-specific branches for GNN-TARnet was set to the value obtained by the random search hyperparameter optimization algorithm implemented in the Keras Tuner library [50]. We set the search space for the parameters as follows. The number of hidden units in the treatment-specific branches n_{hid}^0 and n_{hid}^1 can take values between 16 and 256 in steps of 16. The number of fully connected layers, n_{fc}^0 and n_{fc}^1, was varied between 2 and 10 in steps of 1. The hyperparameters for the IHDP_A, IHDP_B, JOBS and SUM datasets are listed in Table 1, where n_{hid}^{gnn} is the number of hidden units; and n_{fc}^{gnn} is the number of fully connected layers in the GNN block; bs is the batch size; and lr is the learning rate.

We used a Stochastic Gradient Descent (SGD) [51] optimizer with 0.9 momentum, as we empirically found that it works better for all considered models than the ADAM [52] optimizer on the considered datasets. The graph adjacency matrix A for datasets with unknown graph information, such as IHDP_A, IHDP_B, and JOBS was identified using the LiNGAM implemented in the *LiNGAM* package [53], as well as the PC and GES algorithms realized in the *causalai* library [54].

The neural network parameters were initialized using the values drawn from the normal distribution $\mathcal{N}(0, 0.05^2)$. The optimal parameters for the other models were determined according to the procedure described in [17]. All calculations were performed on a system with the following specifications: NVIDIA GeForce RTX 2060 SUPER, AMD Ryzen 7 3800X 8-core processor, and 32 GB RAM.

D. RESULTS

The comparison results between GNN-TARnet and other models are summarized in Table 2 and Table 3. We can see that the proposed method performs on par with other methods. In the JOBS dataset, no differences were observed between the use of empty or discovered graphs. On the IHDP_A dataset, using causal graph found with LiNGAM method resulted in 28% improvement in $\sqrt{\epsilon_{\text{PEHE}}}$ score computed on the test set compared to using an identity graph. However, on the IHDP_B dataset, we observed that using an identity graph gave 15% better $\sqrt{\epsilon_{\text{PEHE}}}$ test score than using a graph discovered with the LiNGAM method. On the artificial SUM dataset, GNN-TARnet using a real graph significantly outperformed the variant using the identity graph and TARnet approaches when the training set size was less than 64.

The results of GNN-TARnet on existing datasets with masked nodes are listed in Table 4. The $\sqrt{\epsilon_{\text{PEHE}}}$ values are significantly higher than those obtained before masking. The difference between the masking nodes that influence the results with zeros and masking all nodes with zeros is negligible for the JOBS dataset and slightly more evident for the IHDP_A and IHDP_B datasets.

As shown in Fig. 5, for the artificial dataset, the model using a real graph performed better than that using an identity graph when the number of training samples was small. As the training set size increased, the performance of GNN-TARnet using the identity graph improved, eventually outperforming the approach using real graphs. This pattern was observed for the datasets with different numbers of layers. The masked covariates made it significantly more difficult for TARnet to match the performance of GNN-TARnet when the training set contained fewer than 128 samples. It can also be observed that for a higher number of layers, more training samples are required for GNN-TARnet using an identity graph to match the performance of GNN-TARnet using the actual graph.

TABLE 2. Comparison of models $\mathcal{R}_{pol}$ and $\sqrt{\epsilon_{\text{PEHE}}}$ on different datasets. The best-performing models are highlighted in bold.

	JOBS($\mathcal{R}_{pol}$)		IHDP$_{\text{A}}$($\sqrt{\epsilon_{\text{PEHE}}}$)		IHDP$_{\text{b}}$($\sqrt{\epsilon_{\text{PEHE}}}$)	
	Train	Test	Train	Test	Train	Test
SLearner	0.22 ± 0.00	0.23 ± 0.01	0.41 ± 0.03	0.43 ± 0.04	2.15 ± 0.04	2.32 ± 0.06
TLearner	0.23 ± 0.00	0.26 ± 0.01	0.49 ± 0.03	0.50 ± 0.04	2.02 ± 0.04	2.18 ± 0.06
TARnet	0.23 ± 0.00	0.25 ± 0.01	0.37 ± 0.01	0.39 ± 0.01	$\mathbf{1.84 \pm 0.04}$	1.98 ± 0.05
CFR-Wass	0.26 ± 0.00	0.28 ± 0.01	$\mathbf{0.35 \pm 0.03}$	$\mathbf{0.37 \pm 0.04}$	1.97 ± 0.04	2.09 ± 0.05
TEDVAE	$\mathbf{0.20 \pm 0.00}$	$\mathbf{0.23 \pm 0.01}$	0.52 ± 0.03	0.56 ± 0.05	2.02 ± 0.04	2.18 ± 0.06
GANITE	0.24 ± 0.00	0.26 ± 0.01	0.49 ± 0.06	0.51 ± 0.06	2.41 ± 0.05	2.50 ± 0.07
GNN-TARnet (LiNGAM)	0.22 ± 0.00	0.23 ± 0.01	0.40 ± 0.02	0.42 ± 0.03	2.12 ± 0.09	2.29 ± 0.11
GNN-TARnet (GES)	0.22 ± 0.00	0.23 ± 0.01	0.46 ± 0.04	0.48 ± 0.04	2.31 ± 0.08	2.48 ± 0.09
GNN-TARnet (PC)	0.22 ± 0.00	0.23 ± 0.01	0.49 ± 0.06	0.51 ± 0.07	2.78 ± 0.15	2.99 ± 0.17
GNN-TARnet (ident.)	0.22 ± 0.00	0.23 ± 0.01	0.53 ± 0.01	0.54 ± 0.02	1.85 ± 0.04	$\mathbf{1.98 \pm 0.05}$

TABLE 3. Comparison of models ϵ_{ATT} and ϵ_{ATE} on different datasets. The best-performing models are highlighted in bold.

	JOBS(ϵ_{ATT})		IHDP$_{\text{A}}$(ϵ_{ATE})		IHDP$_{\text{b}}$(ϵ_{ATE})	
	Train	Test	Train	Test	Train	Test
SLearner	0.22 ± 0.00	0.23 ± 0.01	0.09 ± 0.01	0.10 ± 0.03	0.24 ± 0.03	0.28 ± 0.04
TLearner	0.15 ± 0.00	0.17 ± 0.01	0.09 ± 0.01	0.11 ± 0.01	0.20 ± 0.03	0.25 ± 0.03
TARnet	0.12 ± 0.00	0.14 ± 0.01	0.09 ± 0.01	0.10 ± 0.01	0.20 ± 0.03	0.23 ± 0.03
CFR-Wass	$\mathbf{0.08 \pm 0.02}$	$\mathbf{0.12 \pm 0.02}$	$\mathbf{0.09 \pm 0.01}$	$\mathbf{0.09 \pm 0.01}$	0.32 ± 0.04	0.35 ± 0.05
TEDVAE	0.16 ± 0.01	0.17 ± 0.02	0.09 ± 0.01	0.11 ± 0.01	0.21 ± 0.03	0.26 ± 0.03
GANITE	0.37 ± 0.03	0.37 ± 0.03	0.17 ± 0.02	0.18 ± 0.02	0.38 ± 0.06	0.43 ± 0.06
GNN-TARnet (LiNGAM)	0.12 ± 0.00	0.14 ± 0.01	0.09 ± 0.01	0.10 ± 0.01	0.24 ± 0.02	0.28 ± 0.04
GNN-TARnet (GES)	0.12 ± 0.00	0.14 ± 0.01	0.09 ± 0.01	0.11 ± 0.01	0.24 ± 0.03	0.26 ± 0.04
GNN-TARnet (PC)	0.12 ± 0.00	0.14 ± 0.01	0.09 ± 0.01	0.10 ± 0.01	0.31 ± 0.04	0.34 ± 0.05
GNN-TARnet (ident.)	0.12 ± 0.00	0.14 ± 0.01	0.09 ± 0.01	0.11 ± 0.01	$\mathbf{0.19 \pm 0.02}$	$\mathbf{0.23 \pm 0.03}$

TABLE 4. Results of GNN-TARnet (LiNGAM) on different datasets for cases when all data or only nodes influencing outcomes Y are masked by zero.

	Train	Test
IHDP$_{\text{A}}$ (masked infl. Y)	1.75 ± 0.35	1.75 ± 0.35
IHDP$_{\text{A}}$ (masked ALL)	1.94 ± 0.41	1.89 ± 0.39
IHDP$_{\text{B}}$ (masked infl. Y)	4.57 ± 0.18	4.60 ± 0.20
IHDP$_{\text{B}}$ (masked ALL)	4.67 ± 0.18	4.67 ± 0.21
JOBS (masked infl. Y)	0.30 ± 0.00	0.30 ± 0.00
JOBS (masked ALL)	0.30 ± 0.01	0.30 ± 0.01

V. DISCUSSION

This section provides a comprehensive discussion of the results and implications of our experiments, focusing on the comparison between the use of identity and actual graphs. We also explore the limitations, compare our method with state-of-the-art methods, and provide future directions for our research.

A. ANALYSIS ON EXISTING DATASETS

We begin by examining the performance of GNN-TARnet on the IHDP$_{\text{A}}$ dataset. For this dataset, GNN-TARnet outperformed the model using an identity graph with the graph obtained using the LiNGAM method. However, our method did not produce results as promising as those obtained using CFR-Wass. Contrary to our expectations, GNN-TARnet performs better with an identity graph than with a graph estimated using causal discovery methods for the IHDP$_{\text{B}}$ dataset. This anomaly suggests that the complex nonlinear relationships between covariates and the outcome in the IHDP$_{\text{B}}$ dataset pose a challenge to the causal discovery algorithm and negatively impact the results of GNN-TARnet using the obtained graph. The results also showed that, in cases where estimating the causal graph is challenging, using an identity graph might be an alternative approach.

By design, only a few randomly selected nodes influence the outcomes of IHDP$_{\text{A}}$ and IHDP$_{\text{B}}$. This means that the causal discovery method failed to identify all such nodes;

otherwise, the results after masking nodes influencing the outcomes and all nodes with zeros would have been the same, which is not the case, as shown in Table 4. In the case of the JOBS dataset, masking nodes influencing the outcomes yields almost the same results as masking all nodes with zeros. This indicates that the causal discovery method was able to correctly identify most nodes influencing the outcomes. Therefore, we obtained results similar to those obtained using the TARnet. However, this also means that the other nodes influence neither the outcomes nor the nodes influencing the outcomes. Overall, for all three existing datasets, there were no causal relationships within the features. Thus, the imperfect performance of GNN-TARnet on the existing datasets can be explained by the lack of connectivity within the features.

The results from the existing datasets support our hypothesis that GNN-TARnet can compete with the state-of-the-art methods in causal inference. This is not surprising because our approach takes the structure of the state-of-the-art TARnet by preserving treatment-specific branches and improves it by allowing it to work with causal graphs. However, the results could have been even better if we had known the real adjacency matrices with correctly defined nodes influencing the outcome. The quality of the adjacency matrix plays an important role and can lead to a better or worse performance of the method.

B. DEALING WITH MASKED NODES

For the SUM dataset, GNN-TARnet performed better than TARnet did. This occurred despite the presence of zero-masked nodes, which influenced the outcome. GNN-TARnet makes the values of such nodes to be equal to the sum of their parent nodes. Using summation as the parent aggregation and node update functions, an update step of the node $d_i^{out} = 0$ that influences the outcome for the SUM dataset with one layer is presented below.

$$h_i^{out} = \phi(d_i^{out} + \sum_{j \in \mathcal{N}_i} \psi(d_j)) = \phi(\sum_{j \in \mathcal{N}_i} \psi(d_j)) = \tag{10}$$

$$= w_i^{\phi}(\sum_{j \in \mathcal{N}_i} w_j^{\psi} d_j)) + b_i^{\phi} = \tag{11}$$

$$= \sum_{j \in \mathcal{N}_i} d_j. \tag{12}$$

In Equation (10), h_i^{out} is an updated value of the masked node, d_j is the j-th parent of the node d_i^{out} where j comes from a set $\mathcal{N}_i$ of indices of the parent node, ϕ and ψ are the update and prepare functions from the GNN block (Fig. 3). In (11), w_i^{ϕ} and b_i^{ϕ} are the weight and bias of the update function ψ, and w_j^{ψ} is the weight of the aggregate function ψ. Assuming that weights and biases were identified correctly during the training to be one and zero respectively, in (12), we arrive at the conclusion that updated masked node values are indeed equal to the sum of their parents. As the outcomes of the SUM dataset are computed as a summation or an average value of the nodes influencing the outcomes, knowing the values of such nodes makes ITE estimation trivial. This behavior is particularly prominent when the number of training samples is low. When sufficient data are available, the fully connected layers of the TARnet network can learn any relationship between features and match the performance of the GNN-TARnet.

C. LIMITATIONS

The primary limitation of the proposed method is that it shows promising results only on the artificial dataset. This is because an incorrectly specified graph can drastically reduce its performance, for example, in the case of the IHDP_A or IHDP_B datasets. Thus, if no graph information is available and the training set is sufficiently large, methods that do not rely on graph information may be more beneficial and computationally efficient. It is also crucial to recognize that our approach requires more computational resources than the other methods, which could restrict its use in certain situations. This occurs because the number of parameters increases with an increase in the number of features owing to the feature embedding. However, by computing the outcomes using only a handful of nodes influencing the outcomes, the method can work with relatively large input spaces, which is advantageous compared to other methods and makes it feasible for ITE estimation.

D. FUTURE WORK

Our goal in future work will be to evaluate this method using real data from RCT and other real-life observational studies. Another interesting direction for future research is to explore how to embed causal graphs generation in the training procedure. The ability of GNN-TARnet to handle masked nodes is a notable advantage over the other approaches. In clinical studies that use questionnaires, the final scores are often a composite of the responses to different questions. However, manual calculation of such metrics can be tedious and error-prone. Using GNN-TARnet, we can potentially group covariates based on their association with a specific questionnaire and point them to the masked output node which potentially allows GNN-TARnet to aggregate the final questionnaire score automatically, and after conditioning on it and the scores of other questionnaires, predict the ITE for the entire dataset. This approach is promising for scenarios with few training samples but an abundance of high dimensional features and will serve as a basis for our future research efforts. Reducing the computational burden of the method is also an exciting topic for future work. Creating publicly available datasets with features that are causally dependent on each other and the outcome is another interesting topic that can lead to better data verification and may help to develop better methods using GNNs for causal effect estimation.

VI. CONCLUSION

This study demonstrates how existing knowledge about the causal dependencies of a system can improve ITE prediction. Our results indicate that extending an existing model to

work with DAGs via GNN layers can provide substantial benefits when the training test size is small, and can be used, for example, in clinical settings where information about causal relationships within features is available. This was demonstrated by using an artificial SUM dataset. However, for $IHDP_B$ dataset, when sufficient data are available and finding the causal graph is difficult, one can use TARnet or a GNN-TARnet variant using an identity graph to obtain 15% better performance in $\sqrt{\epsilon_{\text{PEHE}}}$ than the best result of GNN-TARnet (LiNGAM). Overall, we can say that our method is worth considering if the training set size is limited and the causal graph is easy to find or if it is already available from the data provider.

REFERENCES

[1] U. Shalit, "Can we learn individual-level treatment policies from clinical data?" *Biostatistics*, vol. 21, pp. 359–362, Nov. 2019, doi: 10.1093/biostatistics/kxz043.

[2] M. A. Hernan, "A definition of causal effect for epidemiological research," *J. Epidemiology Community Health*, vol. 58, no. 4, pp. 265–271, Apr. 2004.

[3] D. B. Rubin, "Estimating causal effects of treatments in randomized and nonrandomized studies," *J. Educ. Psychol.*, vol. 66, no. 5, pp. 688–701, Oct. 1974.

[4] P. R. Rosenbaum, *Observational Studies*. Cham, Switzerland: Springer, 2002, pp. 1–17.

[5] A. D. Nichol, M. Bailey, and D. J. Cooper, "Challenging issues in randomised controlled trials," *Injury*, vol. 41, pp. S20–S23, Jul. 2010.

[6] J. Pearl, "Causal inference in statistics: An overview," *Statist. Surveys*, vol. 3, no. 1, pp. 1–23, Jan. 2009.

[7] J. Pearl, "On the consistency rule in causal inference: Axiom, definition, assumption, or theorem?" *Epidemiology*, vol. 21, no. 6, pp. 872–875, Nov. 2010, doi: 10.1097/ede.0b013e3181f5d3fd.

[8] D. Westreich and S. R. Cole, "Invited commentary: Positivity in practice," *Amer. J. Epidemiology*, vol. 171, no. 6, pp. 674–677, Feb. 2010, doi: 10.1093/aje/kwp436.

[9] J. Hoogland, J. IntHout, M. Belias, M. M. Rovers, R. D. Riley, F. E. Harrell, K. G. M. Moons, T. P. A. Debray, and J. B. Reitsma, "A tutorial on individualized treatment effect prediction from randomized trials with a binary endpoint," *Statist. Med.*, vol. 40, no. 26, pp. 5961–5981, Nov. 2021.

[10] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini, "The graph neural network model," *IEEE Trans. Neural Netw.*, vol. 20, no. 1, pp. 61–80, Aug. 2009.

[11] J. Zhou, G. Cui, S. Hu, Z. Zhang, C. Yang, Z. Liu, L. Wang, C. Li, and M. Sun, "Graph neural networks: A review of methods and applications," *AI Open*, vol. 1, pp. 57–81, Jan. 2020.

[12] D. B. Rubin, "Causal inference using potential outcomes: Design, modeling, decisions," *J. Amer. Stat. Assoc.*, vol. 100, no. 469, pp. 322–331, Mar. 2005.

[13] S. R. Künzel, J. S. Sekhon, P. J. Bickel, and B. Yu, "Metalearners for estimating heterogeneous treatment effects using machine learning," *Proc. Nat. Acad. Sci. USA*, vol. 116, no. 10, pp. 4156–4165, Mar. 2019.

[14] U. Shalit, F. D. Johansson, and D. Sontag, "Estimating individual treatment effect: Generalization bounds and algorithms," in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 3076–3085.

[15] J. Yoon, J. Jordon, and M. V. D. Schaar, "GANITE: Estimation of individualized treatment effects using generative adversarial nets," in *Proc. Int. Conf. Learn. Represent.*, 2018, pp. 1–24.

[16] W. Zhang, L. Liu, and J. Li, "Treatment effect estimation with disentangled latent factors," in *Proc. AAAI Conf. Artif. Intell.*, May 2021, pp. 10923–10930.

[17] A. Sirazitdinov, M. Buchwald, J. Hesser, and V. Heuveline. (2022). *Review of Deep Learning Methods for Individual Treatment Effect Estimation With Automatic Hyperparameter Optimization*. [Online]. Available: https://www.techrxiv.org/articles/preprint/Review_of_Deep_Learning_Methods_for_Individual_Treatment_Effect_Estimation_with_Automatic_Hyperparameter_Optimization/20448768

[18] H. Wang, J. Fan, Z. Chen, H. Li, W. Liu, T. Liu, Q. Dai, Y. Wang, Z. Dong, and R. Tang, "Optimal transport for treatment effect estimation," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 36, 2024, pp. 1–21.

[19] M. Cuturi and A. Doucet, "Fast computation of Wasserstein barycenters," in *Proc. 31st Int. Conf. Mach. Learn.*, vol. 32, Jun. 2014, pp. 685–693.

[20] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," 2014, *arXiv:1406.2661*.

[21] A. Micheli, "Neural network for graphs: A contextual constructive approach," *IEEE Trans. Neural Netw.*, vol. 20, no. 3, pp. 498–511, Mar. 2009.

[22] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and P. S. Yu, "A comprehensive survey on graph neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 1, pp. 4–24, Jan. 2021.

[23] Z. Xie, W. Lv, S. Huang, Z. Lu, B. Du, and R. Huang, "Sequential graph neural network for urban road traffic speed prediction," *IEEE Access*, vol. 8, pp. 63349–63358, 2020.

[24] A. Derrow-Pinion, J. She, D. Wong, O. Lange, T. Hester, L. Perez, M. Nunkesser, S. Lee, X. Guo, B. Wiltshire, P. W. Battaglia, V. Gupta, A. Li, Z. Xu, A. Sanchez-Gonzalez, Y. Li, and P. Velickovic, "ETA prediction with graph neural networks in Google maps," in *Proc. 30th ACM Int. Conf. Inf. Knowl. Manage.*, Oct. 2021, pp. 3767–3776, doi: 10.1145/3459637.3481916.

[25] D. Xu, Y. Zhu, C. Choy, and L. Fei-Fei, "Scene graph generation by iterative message passing," *Comput. Vis. Pattern Recognit. (CVPR)*, vol. 1, no. 1, pp. 1–25, 2017.

[26] J. Hao, "P-companion: A principled framework for diversified complementary product recommendation," in *Proc. 29th ACM Int. Conf. Inf. Knowl. Manag.*, Oct. 2020, pp. 2517–2524.

[27] D. Jiang, Z. Wu, C.-Y. Hsieh, G. Chen, B. Liao, Z. Wang, C. Shen, D. Cao, J. Wu, and T. Hou, "Could graph neural networks learn better molecular representation for drug discovery? A comparison study of descriptor-based and graph-based models," *J. Cheminformatics*, vol. 13, no. 1, p. 12, Feb. 2021.

[28] N. A. Asif, Y. Sarker, R. K. Chakrabortty, M. J. Ryan, Md. H. Ahamed, D. K. Saha, F. R. Badal, S. K. Das, M. F. Ali, S. I. Moyeen, M. R. Islam, and Z. Tasneem, "Graph neural network: A comprehensive review on non-Euclidean space," *IEEE Access*, vol. 9, pp. 60588–60606, 2021.

[29] C.-T. Li, Y.-C. Tsai, C.-Y. Chen, and J. C. Liao. (2024). *Graph Neural Networks for Tabular Data Learning: A Survey With Taxonomy and Directions*. [Online]. Available: https://synthical.com/article/f8106b43-5a78-4998-a56c-3e60f615856f

[30] P. Zhai, Y. Yang, and C. Zhang, "Causality-based CTR prediction using graph neural networks," *Inf. Process. Manage.*, vol. 60, no. 1, Jan. 2023, Art. no. 103137.

[31] M. Zeč ević, D. Singh Dhami, P. Velickovic, and K. Kersting, "Relating graph neural networks to structural causal models," 2021, *arXiv:2109.04173*.

[32] S. Wein, W. M. Malloni, A. M. Tomé, S. M. Frank, G.-I. Henze, S. Wüst, M. W. Greenlee, and E. W. Lang, "A graph neural network framework for causal inference in brain networks," *Sci. Rep.*, vol. 11, no. 1, p. 8061, Apr. 2021.

[33] Z. Chu, S. L. Rathbun, and S. Li, "Graph infomax adversarial learning for treatment effect estimation with networked observational data," in *Proc. 27th ACM SIGKDD Conf. Knowl. Discovery Data Mining*, Aug. 2021, pp. 176–184.

[34] Á. Parafita and J. Vitria, "Estimand-agnostic causal query estimation with deep causal graphs," *IEEE Access*, vol. 10, pp. 71370–71386, 2022.

[35] Y. Zhang, A. Bellot, and M. van der Schaar, "Learning overlapping representations for the estimation of individualized treatment effects," in *Proc. 23rd Int. Conf. Artif. Intell. Statist.*, vol. 108, Aug. 2020, pp. 1005–1014.

[36] M. Niepert, M. Ahmed, and K. Kutzkov, "Learning convolutional neural networks for graphs," in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2016, pp. 2014–2023.

[37] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl, "Neural message passing for quantum chemistry," in *Proc. 34th Int. Conf. Mach. Learn.*, vol. 70, 2017, pp. 1263–1272.

[38] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," in *Proc. Workshop Deep Learning*, Dec. 2014, pp. 1–12.

[39] P. W. Battaglia et al., "Relational inductive biases, deep learning, and graph networks," 2018, *arXiv:1806.01261*.

[40] Y. Li, D. Tarlow, M. Brockschmidt, and R. Zemel, "Gated graph sequence neural networks," 2015, *arXiv:1511.05493*.

[41] S. Shimizu, P. O. Hoyer, A. Hyvarinen, and A. Kerminen, "A linear non-Gaussian acyclic model for causal discovery," *J. Mach. Learn. Res.*, vol. 7, no. 72, pp. 2003–2030, 2006.
[42] P. Spirtes, C. Glymour, and R. Scheines, *Causation, Prediction, and Search*, 2nd ed., Cambridge, MA, USA: MIT Press, 2000.
[43] D. M. Chickering, "Optimal structure identification with greedy search," *J. Mach. Learn. Res.*, vol. 3, no. 1, pp. 507–554, Mar. 2003.
[44] C. Glymour, K. Zhang, and P. Spirtes, "Review of causal discovery methods based on graphical models," *Frontiers Genet.*, vol. 10, pp. 1–16, Jun. 2019.
[45] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.
[46] J. Brooks-Gunn, F. R. Liaw, and P. K. Klebanov, "Effects of early intervention on cognitive function of low birth weight preterm infants," *J. Pediatrics*, vol. 120, no. 3, pp. 350–359, Mar. 1992.
[47] J. L. Hill, "Nonparametric modeling for causal inference," *J. Comput. Graph. Statist.*, vol. 20, no. 1, pp. 217–240, Jan. 2011.
[48] R. H. Dehejia and S. Wahba, "Causal effects in nonexperimental studies: Reevaluating the evaluation of training programs," *J. Amer. Stat. Assoc.*, vol. 94, no. 448, p. 1053, Dec. 1999.
[49] M. Scutari, "Learning Bayesian networks with thebnlearnRPackage," *J. Stat. Softw.*, vol. 35, no. 3, pp. 1–22, 2010.
[50] T. O'Malley et al. *KerasTuner*. Accessed: Jul. 1, 2024. [Online]. Available: https://github.com/keras-team/keras-tuner
[51] S. Ruder, "An overview of gradient descent optimization algorithms," 2016, *arXiv:1609.04747*.
[52] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. Int. Conf. Learn. Represent.*, 2015, pp. 1–14.
[53] T. Ikeuchi, M. Ide, Y. Zeng, T. N. Maeda, and S. Shimizu, "Python package for causal discovery based on LiNGAM," *J. Mach. Learn. Res.*, vol. 24, no. 14, pp. 1–8, 2023.
[54] D. Arpit, M. Fernandez, I. Feigenbaum, W. Yao, C. Liu, W. Yang, P. Josel, S. Heinecke, E. Hu, H. Wang, S. Hoi, C. Xiong, K. Zhang, and J. C. Niebles, "Salesforce CausalAI library: A fast and scalable framework for causal analysis of time series and tabular data," 2023, *arXiv:2301.10859*.

ANDREI SIRAZITDINOV received the B.S. degree in applied mathematics and informatics from Irkutsk State University, Irkutsk, Russia, in 2016, and the M.S. degree in visual computing from Saarland University, Saarbrücken, Germany, in 2019. He is currently pursuing the Ph.D. degree in computer science with the Data Analysis and Modeling in Medicine Group, Mannheim Institute for Intelligent Systems in Medicine, Heidelberg University, Heidelberg, Germany. He is supervised by Prof. Dr. Jürgen Hesser.

MARCUS BUCHWALD received the B.S. degree in physics from Karlsruhe Institute of Technology, Germany, in 2018, and the M.S. degree in physics, specializing in medical physics, from Heidelberg University, Heidelberg, Germany, in 2021, where he is currently pursuing the integrated Ph.D. degree in physics and computer science with the Data Analysis and Modeling in Medicine Group, Mannheim Institute for Intelligent Systems in Medicine and the Engineering Mathematics and Computing Laboratory, Interdisciplinary Center for Scientific Computing (IWR). He is jointly supervised by Prof. Dr. Jürgen Hesser and Prof. Dr. Vincent Heuveline.

VINCENT HEUVELINE studied mathematics, physics, and computer science from the University of Caen Normandy, France, and the University of Würzburg, Germany. He received the Ph.D. degree in computer science from the Université de Rennes and Institut National de Recherche en Informatique et Automatique (INRIA), in 1997, and the Habilitation degree in mathematics from Heidelberg University, in 2002. Since 2004, he has been a Professor with Karlsruhe University and KIT, until he moved to Heidelberg University, in May 2013. Besides his professorship at IWR, he is currently the Chief Information Officer (CIO) and the Director of the Computing Center, Heidelberg University. He is also the Leader of the Research Group Data Mining and Uncertainty Quantification (DMQ), Heidelberg Institute for Theoretical Studies (HITS gGmbH). Besides lectures in scientific computing, he is strongly involved in teaching IT security, with lectures and dedicated seminars at Heidelberg University. His research interests include uncertainty quantification (UQ) in scientific computing, high-performance and data-intensive computing, and software engineering, with the main application focusing on medical engineering. He serves as a member of the program committee of several international conferences on high-performance and scientific computing. He is widely consulted by the industry concerning the deployment of numerical simulation, cloud computing, and IT security in industrial environments.

JÜRGEN HESSER (Member, IEEE) received the Diploma and Ph.D. degrees in physics from Heidelberg University, in 1989 and 1992, respectively, and the Habilitation degree in mathematics and computer science from the University of Mannheim, Mannheim, Germany, in 1999. From 1998 to 2005, he was a Deputy Professor of medical technology with the University of Mannheim and a Full Professor of experimental radiation oncology from Heidelberg University, in 2007. Since 2019, he has been a Full Data Analysis and Modeling Professor with the Mannheim Institute for Intelligent Systems in Medicine, Medical School Mannheim, Heidelberg University. At Heidelberg University, he is also a Co-Opted Member with the Department of Physics and Astronomy; a member of the Interdisciplinary Center for Scientific Computing (IWR), since 2020; a member of the Board of Directors; a member of the Central Institute for Computer Engineering (ZITI); and a member and a Spokesperson of the CZS Heidelberg Center for Model-Based AI.