



C-HDNet: A Fast Hyperdimensional Computing Based Method for Causal Effect Estimation from Networked Observational Data

Abhishek Dalvi¹ · Neil Ashtekar¹ · Vasant G. Honavar¹

Received: 31 May 2025 / Accepted: 14 July 2025
© The Author(s) 2025

Abstract

We address the problem of estimating causal effects from observational data in the presence of network confounding, a setting where both treatment assignment and observed outcomes of individuals may be influenced by their neighbors within a network structure, resulting in network interference. Traditional causal inference methods often fail to account for these dependencies, leading to biased estimates. To tackle this challenge, we introduce a novel matching-based approach that utilizes principles from hyperdimensional computing to effectively encode and incorporate structural network information. This enables more accurate identification of comparable individuals, thereby improving the reliability of causal effect estimates. Through extensive empirical evaluation on multiple benchmark datasets, we demonstrate that our method either outperforms or performs on par with existing state-of-the-art approaches, including several recent deep learning-based models that are significantly more computationally intensive. In addition to its strong empirical performance, our method offers substantial practical advantages, achieving nearly an order-of-magnitude reduction in runtime without compromising accuracy, making it particularly well-suited for large-scale or time-sensitive applications.

Keywords Causal inference · Network data · Hyperdimensional computing · Low-latency inference

1 Introduction

Many techniques have been proposed to estimate causal effects given observational data (Shalit et al. 2017; Shi et al. 2019; Johansson et al. 2016; Clivio et al. 2022; Stuart 2010) (for additional context and a concise overview, please refer to Igelström et al. (2022), Pearl (2000) and Neal (2020)). Despite the effectiveness of such techniques, only a handful of approaches consider the problem setting in which individuals are connected in a network structure (Guo et al. 2020b, a, c; Veitch et al. 2019). We focus on the problem of estimating individual treatment effects (ITEs) in this setting.

Networked observational data presents an unique opportunity to enhance our understanding of ITEs and plays a pivotal role for informed decision-making in various domains. For example, considering a city or country suffering from an outbreak of a contagious disease. Social connections between individual can be represented as a network, and causal inference becomes crucial in discovering the working of disease transmission. Understanding the biological processes, immune responses, and social interactions within the network sheds light on disease dynamics, as described in Taie and Kadry (2017). Through causal inference, we seek to understand the latent characteristics of disease transmission by identifying significant nodes (individuals) and their network connections. This understanding is crucial for creating tailored intervention strategies, such as vaccination or isolation, in order to disrupt causal pathways and mitigate the spread of the disease within the network. In this example, ignoring the network information in estimating causal effects introduces confounding bias. Incorporating additional information—in this case, social network structure—can avoid unobserved confounding, as described in existing work Guo et al. (2020a, 2020c, 2020b); Veitch et al. (2019).

✉ Abhishek Dalvi
abd5811@psu.edu

✉ Neil Ashtekar
nca5096@psu.edu

✉ Vasant G. Honavar
vuh14@psu.edu

¹ Department of Computer Science and Engineering, The Pennsylvania State University, University Park 16802, PA, USA

Traditional approaches to causal effect estimation rely on straightforward and efficient techniques such as matching (Stuart 2010). Recently, advanced techniques based on deep learning have become popular due to their ability to learn complex representations from data. However, deep learning techniques are computationally expensive, include many tunable hyperparameters, and require a large amount of training data to be effective (Thompson et al. 2021).

Recently, Hyperdimensional (HD) Computing (Kanerva 1992, 2009) has emerged as a promising alternative to deep learning. HD computing does not require iterative optimization for training and instead relies on simple, fast, operations in a high-dimensional space. Data is represented using hyperdimensional, bipolar vectors in $\{-1, 1\}^\beta$, or as multi-bit vectors, where the dimensionality is typically $\beta \approx 10,000$.

In this work, we propose *C-HDNet: a Causal HD computing technique for effect estimation given Network observational data*. To the best of our knowledge, this is among the first approaches that leverage Hyperdimensional Computing for causal inference in settings where both individual-level features and the neighbors of an individual in a network contribute to confounding when estimating the causal effect. The key novelty of C-HDNet lies in its design and algorithm: its ability to encode network relationships and individual covariates into HD vectors, which can then be used to estimate the true causal effect. The primary advantage of our approach over existing methods is that it requires no iterative optimization and operates as a one-pass learning algorithm—i.e., it involves computing specific steps without the need for repetitive training, distinguishing it from deep learning approaches. Hence, it is compute-friendly and provides fast inference while remaining competitive with state-of-the-art models, as demonstrated by our extensive experiments.

The remainder of the paper is organized as follows. Section 2, provides background information on causal inference and HD computing. Section 3 provides precise problem formulation. Section 4 describes our proposed algorithm. Section 5 presents results of experiments that compare our algorithm against state-of-the-art baselines. Finally, Sect. 6 concludes the paper.

2 Background

2.1 Causal inference framework

The frameworks developed by Rubin (2005) and Pearl (2000) offering distinct yet complementary foundations for causal modeling and causal effect estimation. Rubin's framework focuses on counterfactual reasoning to estimate causal effects while Pearl's causal Bayesian networks and DAGs provide a graphical and algorithmic framework for causal

inference. Both frameworks have significantly advanced our ability to infer causal relationships from observational data, each offering complementary tools and insights to address different aspects of the causal inference problem.

In this work we follow Rubin's *potential outcomes* framework. We let T_i denote the treatment assignment for individual i . We assume treatments are binary: T_i can take values $T = 1$ or $T = 0$, but not both. Therefore, calculating the effect of treatment on an individual i requires comparing the outcome under treatment $Y_i^{T=1}$ with the control outcome $Y_i^{T=0}$, of which only one of is observed in the data. We call the observed outcome the factual outcome, and the unobserved outcome the counterfactual outcome. The Individual Treatment Effect (ITE) for individual i is defined as:

$$\text{ITE}_i = Y_i^{T=1} - Y_i^{T=0}.$$

The Average Treatment Effect (ATE) is defined as the expectation over individual treatment effects.

$$\text{ATE} = \mathbb{E}[Y^{T=1} - Y^{T=0}] = \mathbb{E}[Y^{T=1}] - \mathbb{E}[Y^{T=0}].$$

Randomized Control Trials (RCTs) are the gold standard for estimating causal effects. Random assignment ensures that all variables, known and unknown, are equally distributed between the groups in expectation. This setup simplifies the calculation of the Average Treatment Effect (ATE). Unlike observational studies where differences in outcomes between treated and untreated groups might be confounded by other factors, in an RCT, the ATE is essentially the expected difference between the observed outcomes of the treatment and control groups. Therefore, in RCT, estimating the causal effect of treatment reduces to assessing the association between treatment and outcome.

However, RCT are not always feasible due to their high cost or ethical concerns. Hence, there is much interest in methods for reliably estimating causal effects from observational data. The following assumptions suffice for estimating causal effects from observational data (Rosenbaum and Rubin 1983; Cox 1958; Rubin 1980):

1. *Ignorability*, which implies that the treatment assignment T is independent of the potential outcomes $Y^{T=0}$ and $Y^{T=1}$ given the covariates \mathbf{X} . This means that $(Y^{T=0}, Y^{T=1}) \perp\!\!\!\perp T | \mathbf{X} = \mathbf{x}$.
2. *Positivity/Overlap*, indicating that the propensity scores should be between 0 and 1, i.e., $0 < P(T = 1 | \mathbf{X} = \mathbf{x}) < 1$ for all values \mathbf{x} of the covariates \mathbf{X} .
3. *Non-interference*, $Y_i(T_1, \dots, T_i) = Y_i(T_i)$, implying that the outcome of any individual is unaffected by the treatment of all other individuals.
4. *Consistency*, for all t , if $T_i = t$, then $Y_i(t) = Y_i$, ensuring that the observed effect of the assigned treatment is equal to the potential outcome of that treatment and

is necessary for estimating causal effects from observational data.

Under these conditions, it can be shown that:

$$\text{ATE} = \mathbb{E}_{\mathbf{x}}[\mathbb{E}[Y|T = 1, \mathbf{X} = \mathbf{x}] - \mathbb{E}[Y|T = 0, \mathbf{X} = \mathbf{x}]].$$

To estimate causal effects, one must account for potential confounding due to covariates through *covariate adjustment*. One of the simplest techniques for adjusting covariates is *matching*, where the goal is to find a nearly identical “twin” for each individual with the opposite treatment status. Numerous matching techniques have been proposed, based on nearest neighbors, propensity scores (Rubin and Thomas 1996; King and Nielsen 2019) and balancing scores (Rosenbaum and Rubin 1983). A comprehensive overview of matching techniques can be found in Stuart (2010).

Recently, there has been growing interest in deep neural networks for covariate adjustment due to their predictive power. Techniques such as CFRnet (Johansson et al. 2016) learn latent representations for the covariates, minimizing the distributional differences between control and treatment populations. Dragonnet (Shi et al. 2019) leverages Rubin’s lemma (Rubin and Thomas 1996) and makes use of propensity scores with targeted regularization, while borrowing concepts from Shalit et al. (2017) and Chernozhukov et al. (2018). These methods aim to mitigate confounding bias within the original covariates when under binary treatments.

Overall, the above covariate adjustment techniques rely on estimating the outcome using conditional outcome modeling estimators, i.e., $\hat{\mu}(\mathbf{x}, 1) \approx \mathbb{E}[Y | T = 1, \mathbf{X} = \mathbf{x}]$ and $\hat{\mu}(\mathbf{x}, 0) \approx \mathbb{E}[Y | T = 0, \mathbf{X} = \mathbf{x}]$. The estimator $\hat{\mu}$ can range from a simple model such as Nearest Neighbors regressor to a more complex model such as a deep neural network.

2.2 Hyperdimensional computing

Hyperdimensional Computing (Kanerva 1992, 2009; Neubert et al. 2019), offers an attractive alternative to deep learning methods. It represents features as extremely high-dimensional binary or bipolar vectors—often around 10,000 dimensions—on which simple element wise operations are performed for tasks including learning and inference. Unlike deep learning, which relies on iterative minimization of a loss function, HD computing follows predefined steps to map data in a single pass to a high-dimensional space.

In this work, we employ either transformed/hashed or randomly generated bipolar hyperdimensional vectors, denoted as $\{-1, 1\}^\beta$ with $\beta = 10,000$. A collection of such vectors is subsequently used and transformed through hyperdimensional operations to produce multi-bit representations used for

downstream inference. To measure similarity between hyper-vectors, we use a distance metric $d_{\mathcal{H}}$: for bipolar vectors, this is typically the Hamming distance; for multi-bit hyper-vectors, the L2 distance is used. In both cases, cosine similarity may also serve as an alternative similarity measure.

A key property of randomly generated bipolar hyper-vectors is their near-orthogonality: the similarity between any two such vectors, \mathbf{a} and \mathbf{b} , is $d_{\mathcal{H}}(\mathbf{a}, \mathbf{b}) \approx 0.5$ with high probability. This orthogonality, along with the way HD computing distributes information across all dimensions through vector operations, makes hyperdimensional computing an appealing framework for machine learning (Kanerva 1992). Its inherent robustness to noise and support for content-based retrieval offer advantages over traditional deep learning approaches (Neubert et al. 2019). In our study, we make use of the following HD computing operations:

- *Binding* \otimes : is dimension-wise multiplication operation between HD vectors. The resulting vector is dissimilar from the original inputs. For example, given two randomly sampled bipolar vectors \mathbf{a} and \mathbf{b} , if we compute $\mathbf{c} = \mathbf{a} \otimes \mathbf{b}$, then the distances between the result and the original vectors satisfy $d_{\mathcal{H}}(\mathbf{c}, \mathbf{a}) \approx 0.5$ and $d_{\mathcal{H}}(\mathbf{c}, \mathbf{b}) \approx 0.5$, indicating dissimilarity/orthogonal nature for the Binding operation.
- *Bundling* \oplus : is a dimension-wise operation between HD vectors that performs addition across corresponding dimensions. Bundling ensures that the resulting vector, derived from sampled HD vectors within a set, is similar to all the sampled vectors while being dissimilar to non-sampled HD vectors. Further, Bundling is both associative and commutative, meaning: $\mathbf{a} \oplus (\mathbf{b} \oplus \mathbf{c}) = (\mathbf{a} \oplus \mathbf{b}) \oplus \mathbf{c} = (\mathbf{c} \oplus \mathbf{a}) \oplus \mathbf{b}$. Bundling is often used as an aggregate representation of a set of hyper vectors. In some literature, Bundling is performed as signed addition, while in this work we use dimension-wise addition, following the approach used by Kang et al. (2022).

Combining Bundling and Binding enables concise representations of data with complex features. For instance, consider a dataset containing information about people. Consider one individual “Alice”, who is of age 25, height of 170, and weight of 120. Alice’s features can be represented using an information-preserving hypervector as follows: $Alice_{HV} = (Age_{HV} \otimes 25_{HV}) \oplus (Height_{HV} \otimes 170_{HV}) \oplus (Weight_{HV} \otimes 120_{HV})$. The other individuals in the dataset can be encoded in a similar manner. For a more in-depth overview please refer to Kanerva (2009) and Neubert et al. (2019).

3 Problem statement

We are given a dataset with N individuals, with each individual $i \in \{1, \dots, N\}$ represented by a d -dimensional feature vector $\mathbf{x}_i \in \mathbb{R}^d$. We assume treatments are binary: $T_i \in \{0, 1\}$ denotes the observed treatment, and $Y_i^{(T_i)} \in \mathbb{R}$ denotes the corresponding observed (factual) outcome for individual i . We use \mathbf{A} to denote the adjacency matrix of the network, which represents connections between individuals. We assume that the network is undirected and that all edges are unweighted, i.e., $A_{ij} = A_{ji}$ and $\mathbf{A} \in \{0, 1\}^{N \times N}$.

In this work, we assume that the treatment assignment and outcomes for individual i are causally influenced by $\mathcal{N}^k(i)$, which represents the k -hop neighbors of node i . In this setting, some of the assumptions listed in Sect. 2 are violated:

- *Violation of the Non-interference Assumption:*
 $Y_i(T_1, \dots, T_i, \dots, T_N) \neq Y_i(T_i)$
- *Violation of the Conditional Ignorability Assumption:*
 $Y_i(0), Y_i(1) \not\perp\!\!\!\perp T_i | \mathbf{x}_i$

Given that we assume an individual’s treatment assignment and outcomes are causally influenced by \mathbf{x}_i and $\mathcal{N}^k(i)$, we posit the existence of a latent variable \mathbf{Z} acting as a confounder. This latent variable serves as a proxy for both the individual covariates \mathbf{X} and the network structure \mathbf{A} . Conditioning on \mathbf{Z} mitigates the confounding effects of treatment \mathbf{T} on outcome \mathbf{Y} .

Therefore, given observational data $\mathcal{D} = \left(\left\{ (\mathbf{x}_i, T_i, Y_i^{(T_i)}) \right\}_{i=1}^N, \mathbf{A} \right)$, our objective is to represent the latent variable $\mathbf{Z} = f(\mathbf{X}, \mathbf{A})$ as function of individual features \mathbf{X} and network connections \mathbf{A} . This ensures that the condition $Y_i^{(1)}, Y_i^{(0)} \perp\!\!\!\perp T_i | \mathbf{z}_i$ or $Y_i^{(1)}, Y_i^{(0)} \perp\!\!\!\perp T_i | (\mathbf{x}_i, \mathcal{N}^k(i))$ is satisfied $\forall i \in \{1 \dots N\}$, enabling unbiased estimation of individual treatment effects. Refer to Fig. 1 for a causal diagram illustrating this problem setting.

4 C-HDNet

Conventional covariate adjustment techniques—which condition only on \mathbf{X} —are inadequate for our problem setting as they fail to address confounding from \mathbf{Z} . Techniques proposed by Guo et al. (2020b, 2020a, 2020c) address this issue by modeling the latent variable \mathbf{Z} , which serves as a proxy for both \mathbf{X} and \mathbf{A} , using Graph Neural Networks (Kipf and Welling 2017; Veličković et al. 2018; Hamilton et al. 2017). These networks are used because of their strong predictive

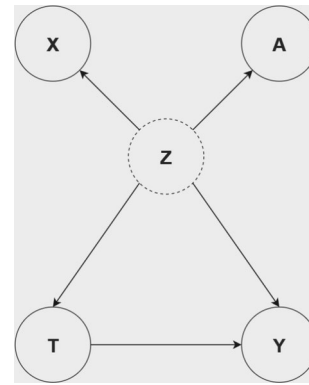


Fig. 1 The causal diagram for the networked observational data problem setting. The covariates or features for nodes/individuals are represented by \mathbf{X} , the network structure is represented by \mathbf{A} . \mathbf{Z} is a latent representation, thus represented by a dotted circle. Ideally, conditioning on \mathbf{Z} should deconfound the causal effect of treatment \mathbf{T} on outcome \mathbf{Y}

performance across a wide range of applications (Chen et al. 2020; Wu et al. 2020).

Our primary goal is to design a model that is both low-latency and performant. To achieve this goal, we first map the original covariates to a latent representation which incorporates network information, then we perform nearest neighbor matching.

4.1 Mapping covariates to hyperdimensional representations

To make use of HD computing, we must first represent our data with hyperdimensional vectors. To do so, we start by mapping our covariates $\mathbf{x}_i \in \mathbb{R}^d$ using Random Hyperplane Tessellations (Plan and Vershynin 2014; Dirksen et al. 2022; Dalvi et al. 2024). Specifically, we map our covariates to hyperdimensional vectors \mathbf{r}_i using the following equation:

$$\mathbf{r}_i = \text{sign}(\mathbf{Q}\mathbf{x}_i + \Gamma). \tag{1}$$

Here, Γ is sampled from a uniform distribution over $[-\lambda, \lambda]^\beta$, and \mathbf{Q} is a matrix in $\mathbb{R}^{\beta \times d}$ with rows drawn from a d -dimensional normal distribution.

Previous research by Dalvi et al. (2024) demonstrated that if $Y_i^{(1)}, Y_i^{(0)} \perp\!\!\!\perp T_i | \mathbf{x}_i$, then the transformation in Eq. 1 serves as a valid approximate balancing score (Rosenbaum and Rubin 1983). Specifically, Dalvi et al. (2024) showed that $\forall i \in \{1 \dots N\}, (Y_i^{(1)}, Y_i^{(0)}) \perp\!\!\!\perp T_i | \mathbf{x}_i \implies (Y_i^{(1)}, Y_i^{(0)}) \perp\!\!\!\perp T_i | \mathbf{r}_i$ approximately holds with high probability when β is sufficiently large and the true propensity score function $e(\mathbf{x}_i) = \mathbb{E}[T = 1 | \mathbf{x}_i]$ is smooth. This conclusion is based on the findings that the mapping from \mathbf{x}_i to \mathbf{r}_i (i) preserves distances with extremely high probability when β is sufficiently large and (ii) provides no additional information regarding

treatment assignment, thereby maintaining conditional independence.

In our setting, the network itself may act as a confounder. This could result in a scenario where $Y_i(1)$ and $Y_i(0)$ are not independent of T_i given \mathbf{r}_i . Regardless, since we assume that $Y_i^{(1)}, Y_i^{(0)} \perp\!\!\!\perp T_i \mid (\mathbf{x}_i, \mathcal{N}^k(i))$; we can assert that $Y^{(1)}, Y^{(0)} \perp\!\!\!\perp T \mid (\mathbf{r}_i, \mathcal{N}^k(i))$. Again, this is because the mapping in Eq. 1 preserves ignorability.

To enable matching to deconfound the effect of the network structure, we design a function that utilizes HD computing. This function incorporates network information using hypervectors, resulting in a latent representation which can then be used for matching.

4.2 Incorporating network information

We now describe how network structure is incorporated in our HD representation. We make use of the RelHD method proposed in Kang et al. (2022). RelHD constructs a representation \mathbf{z}_i for node i as follows:

$$\mathbf{z}_i = (\psi_0 \otimes \mathbf{r}_i) \oplus (\psi_1 \otimes \mathbf{h}_i^{1\text{-hop}}) \oplus (\psi_2 \otimes \mathbf{h}_i^{2\text{-hop}}) \tag{2}$$

where ψ_0, ψ_1 , and ψ_2 are randomly-sampled bipolar hypervectors with fixed values across all nodes $i \in \{1, \dots, N\}$. The \mathbf{h}_i terms are constructed as:

$$\mathbf{h}_i^{1\text{-hop}} = \bigoplus_{j \in \mathcal{N}^1(i)} \mathbf{r}_j$$

$$\mathbf{h}_i^{2\text{-hop}} = \bigoplus_{j \in \mathcal{N}^1(i)} \mathbf{h}_j^{1\text{-hop}}$$

In this encoding, $\mathbf{h}_i^{1\text{-hop}}$ aggregates the 1-hop neighborhood information of node i , while $\mathbf{h}_i^{2\text{-hop}}$ aggregates the 2-hop neighborhood information of node i by propagating the previously-aggregated 1-hop information. The $\mathbf{r}_i, \mathbf{h}_i^{1\text{-hop}}$, and $\mathbf{h}_i^{2\text{-hop}}$ vectors are binded to the ψ_0, ψ_1 , and ψ_2 vectors in order to indicate relative network position. The resulting terms are bundled, aggregating node-level and neighborhood-level information. Note that this encoding method inherently accommodates sparse networks and disconnected nodes. For example, if there exist a node p which is disconnected, the latent representation of node p will be $\mathbf{z}_p = (\psi_0 \otimes \mathbf{r}_p)$, since $\mathbf{h}_p^{1\text{-hop}} = \mathbf{h}_p^{2\text{-hop}} = [0]^{10000}$, since $\mathcal{N}^1(p) = \emptyset$

Our inclusion of 1-hop and 2-hop neighbor information is motivated by the literature on graph neural networks (GNNs). Including information from 1-hop and 2-hop neighbors is typically sufficient for performing downstream classification or regression tasks while avoiding the GNN *oversmoothing* problem (Chen et al. 2020). Oversmoothing occurs when samples with meaningfully different labels are mapped to similar latent representations. Incorporating information

from distant nodes (beyond 2-hop) can result in oversmoothing, which may limit predictive performance.

We hypothesize that our constructed representation \mathbf{z}_i possesses sufficient information to approximately satisfy conditional ignorability, i.e., $Y^{(1)}, Y^{(0)} \perp\!\!\!\perp T \mid \mathbf{Z}$, given the causal model in Fig. 1. Assuming this condition holds, matching can be performed to faithfully estimate treatment effects.

The primary advantage of our approach over existing methods is that ours requires *no training* and operates as a one-pass learning algorithm. Unlike state-of-the-art deep learning techniques, our approach does not involve iterative optimization (i.e. no backpropagation or gradient descent), resulting in far greater computational efficiency.

4.3 Nearest neighbor matching for outcome prediction

Matching, as discussed in Stuart (2010), is a method used to mitigate bias from confounding variables when estimating causal effects in observational data. In our approach, we perform matching using the latent representation \mathbf{Z} in order to account for the covariates as well as the network. This method effectively reduces bias from observed confounders and operates efficiently through weighted k -nearest neighbors regression. Note that we use m rather than k to denote the number of neighbors and avoid abuse of notation.¹

To calculate the counterfactual outcome for an individual i , we use $m\text{-NN}_1$ and $m\text{-NN}_0$, which represent the m -nearest neighbors from the treatment and control groups, respectively. The outcomes are computed as follows:

$$\hat{Y}_i^1 = \sum_{j \in m\text{-NN}_1(\mathbf{z}_i)} w_j Y_j;$$

$$\hat{Y}_i^0 = \sum_{j \in m\text{-NN}_0(\mathbf{z}_i)} w_j Y_j$$

Here, w_j denotes weights assigned based on distance (i.e., nearer neighbors are assigned more weight) under the constraint that $\sum_j w_j = 1$.

5 Experiments

We proceed to describe the data sets used, the experimental setup, the state-of-the art baselines used in our experiments, and results of comparison of C-HDNet with state-of-the-art methods for causal effect estimation from observational data in the presence of network interference. We also analyze the effect of \$k\$, the number of hops used to incorporate

¹ Recall that $\mathcal{N}^k(i)$ represents the k -hop neighbors of node i in the remainder of the paper.

Table 1 Summary of BlogCatalog and Flickr graph datasets

Statistic	BlogCatalog	Flickr
Number of nodes	5,196	7,575
Dimensionality of features	2,198	1,205
Number of links	171,743	239,738
Average node degree	66.11	63.30
Median node degree	49	25
Node degree (10th percentile)	22	7
Node degree (90th percentile)	134	147

information from network neighbors of each node into its HD representation. The codebase for our experiments can be accessed using the following link: <https://github.com/Abhishek-Dalvi410/C-HDNet>.

5.1 Datasets

As observational data does not include counterfactual outcomes, synthetic or semi-synthetic data is typically used in the evaluation of causal effect estimation techniques. We perform evaluations on two datasets—BlogCatalog and Flickr—used in Guo et al. (2020a, 2020b). An overview of the datasets is provided in Table 1; the spread of the node degrees—captured by the average, median, and percentile values—highlight the realistic nature of these datasets, showcasing sparsely and densely connected regions.

- *BlogCatalog*: In this dataset, each node represents a blogger, with links indicating social relationships between bloggers. The node features includes processed bag-of-words representations derived from keywords extracted from blogger descriptions. The treatment or intervention pertains to whether a blogger’s content receives more views on mobile devices ($T = 1$) or desktops ($T = 0$). The outcomes indicate the opinions of readers for each blogger.
- *Flickr*: This dataset represents an online social network where users share images and videos. Nodes are users and each edge signifies a social connection (friendship) between a pair of users. Users’ features are constructed from a list of interest tags on images and videos. Following the same methodology and assumptions as applied to the BlogCatalog dataset, user’s content is viewed on mobile devices ($T = 1$) or desktops ($T = 0$). The outcomes indicate the opinions of readers on each user profile.

We adopt the data generation process described in Guo et al. (2020b, 2020a) for the setting where a user’s features and those of their neighbors influence treatment assignment and

reader opinions (see Fig. 1). We want to know how receiving more views on mobile devices, compared to desktops, impacts reader opinions—an exploration of individual treatment effects in this context. We evaluate the three versions of each dataset provided by Guo et al. (2020b, 2020a) with the confounding factor for the individual (node) covariates set to $\kappa_0 = 10$, while the confounding factor for the covariates of the 1-hop neighbors are set to $\kappa_1 = \{0.5, 1, 2\}$. The larger the value of the confounding factor, the stronger influence of confounding from neighboring nodes on the overall effect.

Additionally, we include in our analyses, confounding effects from 2-hop neighbors, denoted as κ_2 (refer to Eq. (11) in Guo et al. (2020b), where an additional term for 2-hop neighbors is included). To better reflect real-world scenarios, we randomize the confounding factors for each node— $\{\kappa_0^i, \kappa_1^i, \kappa_2^i\}$ sampling them a Uniform distribution $\mathcal{U}; \forall i \in \{1 \dots N\}$. In contrast, Guo et al. (2020b, 2020a) considered the confounding factors to have same values across all nodes. For the Flickr dataset, we sample $\kappa_1^i \sim \mathcal{U}(0.5, 3)$, while for the BlogCatalog dataset, $\kappa_1^i \sim \mathcal{U}(0.5, 1)$. In both datasets $\kappa_0^i \sim \mathcal{U}(5, 10)$, $\kappa_2^i \sim \mathcal{U}(0.01, 0.05)$.

5.2 Evaluation metrics

We evaluate C-HDNet against other existing methods using two primary categories of performance metrics: (1) the error in estimated causal effects compared to their true values, and (2) the time required for training and inference.

For the assessment of causal effect estimation error; we use the error on the Average Treatment Effect denoted by ϵ_{ATE} and the Rooted Precision in Estimation of Heterogeneous Effect (PEHE) error (Hill 2011) denoted by ϵ_{PEHE} . These errors are estimated as follows:

$$\epsilon_{ATE} = \left| \left(\frac{1}{N} \sum_{i=1}^N \widehat{ITE}_i \right) - \left(\frac{1}{N} \sum_{i=1}^N ITE_i \right) \right|$$

$$\epsilon_{PEHE} = \sqrt{\frac{1}{N} \sum_{i=1}^N (\widehat{ITE}_i - ITE_i)^2}$$

where and $ITE_i = Y_i^1 - Y_i^0$ denotes the true Individual Treatment Effect and $\widehat{ITE}_i = \widehat{Y}_i^1 - \widehat{Y}_i^0$ denotes the Estimated Individual Treatment Effect each for the i^{th} sample.

Furthermore, we conduct our experiments in two contexts: (1) *In-sample*, where we estimate the causal effect using the available factual outcome, and (2) *Out-of-sample*, where our goal is to estimate the causal effect in the absence of both the potential outcomes. For all our experiments, we split our data such that 20% of the samples are used for

out-of-sample evaluation and the remaining 80% of the samples are used for in-sample evaluation.

5.3 Baseline methods

We compare C-HDNet against the following baselines:

- TARNet (Shalit et al. 2017; Johansson et al. 2016): Treatment-Agnostic Representation Networks learns representations of confounders by transforming the original features into a latent space using a neural network. This network is then split into two parts to predict potential outcomes. TARNet is trained to minimize the error in inferred factual outcomes while also working to reduce the discrepancy across confounder representations over the treated and controlled groups. In this work, we use the maximum mean discrepancy (MMD) variant for balancing penalties.
- DRGNet (Shi et al. 2019): DragonNet similar to TARNet in terms of its network architecture. However, the network is trained to minimize the error in inferred factual outcomes while forcing latent representations to be suggestive of the propensity score using targeted regularization, based on non-parametric estimation theory.
- C-VAE (Louizos et al. 2017): Causal Effect Variational Autoencoder is a neural network latent that employs latent variable modeling to concurrently estimate the hidden latent space that summarizes the confounders as well as the causal effect. This method is based on Variational Autoencoders (VAE) Kingma (2013), which adhere to the causal structure of inference using proxies.
- C-Forest (Wager and Athey 2018): Causal Forest is a non-parametric method for estimating heterogeneous treatment effects which extends the widely used Random forest algorithms to incorporate causal effect estimation.
- BART (Hill 2011; Chipman et al. 2010): Bayesian Additive Regression Trees are a Bayesian nonparametric modeling procedure that allows for flexible modeling and accommodates complex relationships between confounders, treatment, and outcomes to effectively estimate causal effects.
- NetDeconf (Guo et al. 2020a): Network Deconfounder shares a similar architecture to TARNet, and is trained using the same objective function. However, it utilizes graph neural networks to map covariates into latent representations, which incorporate network information in the latent space to assist in network deconfounding effects.

5.4 Experimental setup

All experiments were conducted on Google Colab Pro using the high RAM version, with all models utilizing only CPU and no GPUs for computation. For C-HDNet, we chose $\beta = 10,000$ dimensions for our hyperdimensional representations, as this number has been shown to be sufficient for the properties of HD computing to hold (Kanerva 2009; Neubert et al. 2019; Dalvi et al. 2024). We used the KNN regressor from scikit-learn for matching, keeping all parameters at their default settings including the number of nearest neighbors $m = 5$. Furthermore, we set $\Gamma = 0$ in Eq. (1). According to the findings of Dirksen et al. (2022), the parameter Γ is necessary to distinguish or assign different binary representations to data points that lie along the same ray or concentrated along a line. However, our datasets happen to be well-distributed and scattered, making this distinction unnecessary.

For TARNet and Dragonnet, the architecture consists two layers for the latent representation followed by two layers in each of the two heads for outcome predictions. The hidden units and layers are structured as [512, 256, [128, 128], [128, 128]] for TARNet and [200, 200, [100, 100], [100, 100]] for DragonNet. All loss terms in both models are weighted equally. For CEVAE, we utilized the implementation from the causalml library (Chen et al. 2020), using the hyperparameters specified in the example implementation on the documentation page. In the case of Causal Forest, we set the hyperparameters to include 20 trees, with a split ratio of 0.7, a minimum of 5 observations of each type (treated and untreated) allowed in a leaf, and a maximum tree depth of 20. For BART and NetDeconf, we used the implementations provided by Guo et al. (2020b).

5.5 Comparison with the state-of-the-art

The results under predefined levels of confounding for the BlogCatalog dataset are presented in Tables 2 and 3, corresponding to in-sample and out-of-sample performance, respectively. Similarly, the in-sample and out-of-sample results for the Flickr dataset are shown in Tables 4 and 5. On each simulation, our proposed method (C-HDNet) either outperforms all existing methods or performs similarly to the best-performing existing method. As expected, the methods which do not incorporate network information (TARNet, DRGNet, C-VAE, C-Forest, and BART) generally perform worse than the methods which do incorporate network information (NetDeconf, C-HDNet). Further, the error in effect estimation generally increases as the amount of network confounding (κ_1 , confounding from 1-hop neighbors) increases.

Our results for randomized confounding are given in Table 6. Here, we compare the best performing existing method (NetDeconf) against our proposed method

Table 2 In-sample mean errors and standard errors across 10 simulations of the BlogCatalog dataset generated by Guo et al. (2020b, 2020a)

Method	$\kappa_1 = 0.5$		$\kappa_1 = 1$		$\kappa_1 = 2$	
	ϵ_{ATE}	ϵ_{PEHE}	ϵ_{ATE}	ϵ_{PEHE}	ϵ_{ATE}	ϵ_{PEHE}
TARNet	1.4 ± 0.4	8.4 ± 0.6	2.0 ± 0.3	14.2 ± 0.6	4.4 ± 0.9	28.6 ± 0.6
DRGNet	1.4 ± 0.3	6.8 ± 0.3	1.8 ± 0.3	11.8 ± 0.2	3.9 ± 0.9	24.0 ± 0.3
C-VAE	1.8 ± 0.2	8.5 ± 0.1	5.7 ± 0.5	16.3 ± 0.4	9.4 ± 1.7	32.2 ± 1.2
C-Forest	4.4 ± 1.2	8.3 ± 1.5	3.5 ± 0.7	8.5 ± 1.0	11.9 ± 2.2	22.3 ± 2.6
BART	5.9 ± 1.4	10.4 ± 2.1	6.4 ± 0.9	10.9 ± 0.9	17.4 ± 3.2	28.2 ± 3.0
NetDeconf	0.7 ± 0.2	4.7 ± 0.8	1.3 ± 0.2	4.7 ± 0.5	2.1 ± 0.5	9.1 ± 1.2
C-HDNet	0.6 ± 0.1	2.7 ± 0.1	0.8 ± 0.2	4.1 ± 0.2	2.0 ± 0.4	7.4 ± 0.3

Table 3 Out-of-sample mean errors and standard errors across 10 simulations of the BlogCatalog dataset generated by Guo et al. (2020b, 2020a)

Method	$\kappa_1 = 0.5$		$\kappa_1 = 1$		$\kappa_1 = 2$	
	ϵ_{ATE}	ϵ_{PEHE}	ϵ_{ATE}	ϵ_{PEHE}	ϵ_{ATE}	ϵ_{PEHE}
TARNet	1.7 ± 0.5	10.8 ± 1.4	2.5 ± 0.4	16.4 ± 1.3	4.9 ± 1.2	35.3 ± 2.3
DRGNet	1.3 ± 0.3	7.4 ± 1.2	1.8 ± 0.3	9.7 ± 1.0	3.8 ± 1.0	23.0 ± 2.5
C-VAE	6.0 ± 0.9	11.7 ± 1.5	8.9 ± 1.2	14.4 ± 1.3	21.3 ± 2.0	30.6 ± 2.6
C-Forest	4.5 ± 1.3	8.3 ± 1.5	3.5 ± 0.7	8.6 ± 1.0	11.6 ± 2.3	22.9 ± 2.9
BART	1.5 ± 0.4	3.9 ± 0.5	2.4 ± 0.5	5.5 ± 0.5	8.3 ± 2.9	12.6 ± 2.8
NetDeconf	0.7 ± 0.2	4.7 ± 0.8	1.3 ± 0.2	4.7 ± 0.5	2.1 ± 0.4	9.1 ± 1.3
C-HDNet	0.3 ± 0.1	4.4 ± 0.7	0.7 ± 0.1	4.8 ± 0.4	1.2 ± 0.3	8.8 ± 0.9

Table 4 In-sample mean errors and standard errors across 10 simulations of the Flickr dataset generated by Guo et al. (2020b, 2020a)

Method	$\kappa_1 = 0.5$		$\kappa_1 = 1$		$\kappa_1 = 2$	
	ϵ_{ATE}	ϵ_{PEHE}	ϵ_{ATE}	ϵ_{PEHE}	ϵ_{ATE}	ϵ_{PEHE}
TARNet	2.5 ± 0.6	21.5 ± 1.5	4.4 ± 1.1	40.7 ± 1.9	8.8 ± 1.2	78.6 ± 4.0
DRGNet	1.6 ± 0.5	11.1 ± 0.3	3.1 ± 0.6	21.7 ± 0.6	5.7 ± 1.7	43.8 ± 1.2
C-VAE	0.6 ± 0.1	13.0 ± 0.0	2.6 ± 0.4	24.8 ± 0.5	4.1 ± 1.1	46.4 ± 1.0
C-Forest	3.1 ± 0.7	8.6 ± 0.4	7.9 ± 1.3	16.4 ± 1.0	14.2 ± 2.5	31.0 ± 1.5
BART	3.9 ± 0.4	9.8 ± 0.3	5.1 ± 0.8	16.8 ± 0.8	8.2 ± 1.4	31.4 ± 1.2
NetDeconf	0.7 ± 0.1	4.2 ± 0.2	1.2 ± 0.1	6.0 ± 0.2	2.3 ± 0.2	9.7 ± 0.4
C-HDNet	0.7 ± 0.0	3.1 ± 0.0	1.0 ± 0.0	6.0 ± 0.4	1.4 ± 0.1	11.9 ± 1.6

Table 5 Out-of-sample mean errors and standard errors across 10 simulations of the Flickr dataset generated by Guo et al. (2020b, 2020a)

Method	$\kappa_1 = 0.5$		$\kappa_1 = 1$		$\kappa_1 = 2$	
	ϵ_{ATE}	ϵ_{PEHE}	ϵ_{ATE}	ϵ_{PEHE}	ϵ_{ATE}	ϵ_{PEHE}
TARNet	3.8 ± 0.9	28.4 ± 2.1	6.5 ± 1.7	54.7 ± 3.8	13.2 ± 2.0	99.4 ± 5.8
DRGNet	2.2 ± 0.7	10.2 ± 0.6	3.4 ± 0.7	18.3 ± 1.7	8.5 ± 3.1	36.2 ± 2.5
C-VAE	1.0 ± 0.1	14.3 ± 0.4	2.8 ± 0.4	24.8 ± 1.1	8.3 ± 0.6	40.4 ± 1.4
C-Forest	2.9 ± 0.7	8.8 ± 0.4	7.8 ± 1.3	16.8 ± 1.2	14.3 ± 2.5	32.1 ± 1.9
BART	2.2 ± 0.5	4.8 ± 0.3	2.5 ± 0.9	6.2 ± 0.6	5.3 ± 0.9	9.9 ± 0.8
NetDeconf	0.7 ± 0.1	4.2 ± 0.3	1.2 ± 0.2	6.1 ± 0.3	2.5 ± 0.2	9.8 ± 0.4
C-HDNet	0.5 ± 0.0	4.6 ± 0.3	0.7 ± 0.0	6.9 ± 0.5	0.9 ± 0.1	13.0 ± 2.0

(C-HDNet). We observe similar performance across methods for both the BlogCatalog and Flickr datasets.

As a final comparison against existing methods, we consider computational efficiency (combined runtime

for training and inference) rather than predictive performance. Again, we compare C-HDNet against NetDeconf, as both methods incorporate network information and offer similar predictive performance. Results are illustrated in

Table 6 Mean errors and standard errors across 10 simulations of Flickr and BlogCatalog datasets for random levels of 0-hop, 1-hop, and 2-hop confounding

		In-sample		Out-of-sample	
		ϵ_{ATE}	ϵ_{PEHE}	ϵ_{ATE}	ϵ_{PEHE}
BlogCat	NetDeconf	0.3 ± 0.0	1.9 ± 0.0	0.3 ± 0.0	1.0 ± 0.0
	C-HDNet	0.2 ± 0.0	1.9 ± 0.0	0.2 ± 0.0	1.4 ± 0.0
Flickr	NetDeconf	0.2 ± 0.0	4.0 ± 0.0	0.3 ± 0.0	3.0 ± 0.1
	C-HDNet	0.3 ± 0.0	4.2 ± 0.1	0.2 ± 0.0	3.8 ± 0.1

Fig. 2 Runtime comparison between C-HDNet and NetDeconf across different datasets. BlogCat-R and Flickr-R refer to the datasets generated using randomly sampled 0-hop, 1-hop, and 2-hop confounding factors, while the remaining datasets are from Guo et al. (2020a, b)

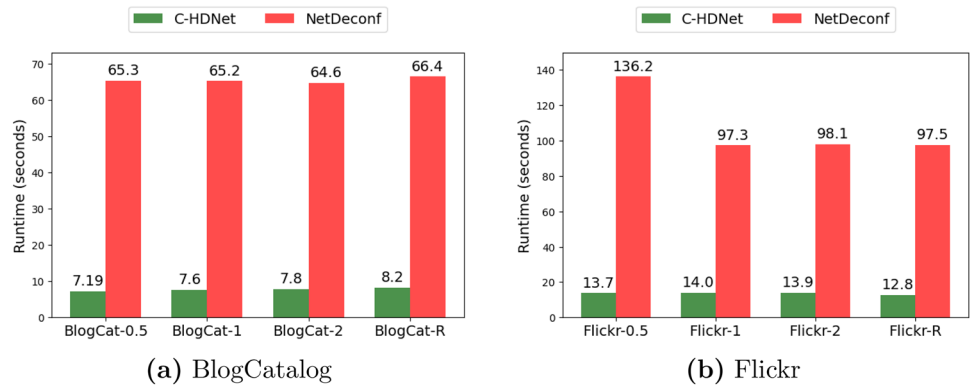
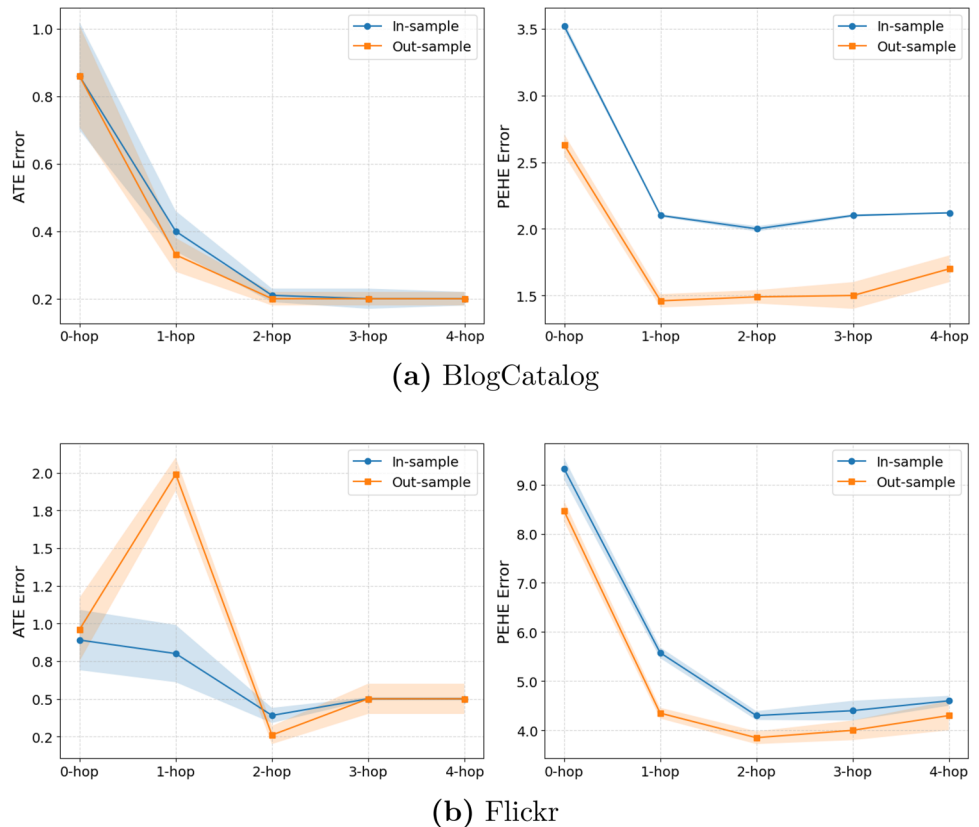


Fig. 3 Mean ATE and PEHE errors of C-HDNet with increasing k -hop neighborhood information. Shaded areas indicate the standard error across 10 simulations



5.6 Evaluating C-HDNet with varying k-Hop configurations

In this section, we design experiments to evaluate how incorporating network information influences the performance of our C-HDNet algorithm. We consider HD representations of covariates under the following configurations: (1) without any network information (0-hop), (2) using up to 1-hop neighbor information, (3) using up to 2-hop neighbor information (i.e., the proposed model), (4) using up to 3-hop neighbor information, and (5) using up to 4-hop neighbor information.

In the 0-hop setting, we set $\mathbf{z}_i = (\psi_0 \otimes \mathbf{r}_i)$, meaning only the node's own covariates are used. For the 1-hop configuration, we remove the ψ_2 and $\mathbf{h}_i^{2\text{-hop}}$ terms from Eq.(2). In contrast, for the 3-hop and 4-hop cases, we extend Eq.(2) by introducing the additional terms $(\psi_3 \otimes \mathbf{h}_i^{3\text{-hop}})$ and $(\psi_4 \otimes \mathbf{h}_i^{4\text{-hop}})$, respectively. The 2-hop setting corresponds to our original proposed model, where \mathbf{z}_i is defined as in Eq. (2).

Results are summarized in Fig. 3. Random values of confounding $\{\kappa_0, \kappa_1, \kappa_2\}$ are used for the Flickr and BlogCatalog datasets—this setup is the same as the one used to generate results for Table 6.

We observe that the predictive performance of C-HDNet is generally strongest when using the full model defined in Eq. (2), which includes up to 2-hop neighborhood information. Performance is weakest when no network information (0-hop) is included. Notably, ATE error on the Flickr dataset deviates from this trend—the worst performance is observed when only 1-hop neighborhood information is used.

Including additional information from 3-hop and 4-hop neighborhoods also results in a performance decline compared to the 2-hop model. This degradation is likely due to the introduction of additional confounding factors or over-smoothing, as discussed in Sect. 4. However, the decline is not as severe as that seen in the 0-hop and 1-hop settings.

In summary, omitting key neighborhood information (as in the 0-hop and 1-hop cases) leads to a substantial drop in performance due to missing confounding signals. On the other hand, incorporating excessive neighborhood depth (3-hop and 4-hop) may introduce noise or irrelevant dependencies, thereby harming predictive accuracy.

6 Summary and discussion

In this work, we have introduced C-HDNet—an algorithm for causal effect estimation given networked observational data. Network data is ubiquitous in the real-world, and incorporating this information when performing causal inference can help avoid confounding and improve causal

effect estimates. Design of the C-HDNet algorithm was motivated by ideas from hyperdimensional computing and graph neural networks. C-HDNet first constructs a latent representation which incorporates network information, then performs matching in the latent space. The results of extensive experiments show that our algorithm either matches or outperforms the state-of-the-art approaches at a fraction of their computational cost.

Looking ahead, several promising directions remain for future research. One important extension is adapting C-HDNet to handle temporal or longitudinal network data, where each node in the network has observations over time, potentially with irregular intervals. This would enable the model to estimate dynamic causal effects in evolving networks.

Another limitation of the current work has to do with the lack of uncertainty quantification. At present, we provide only point estimates of causal effects without offering confidence intervals or other measures of uncertainty. Standard errors derived from repeated simulations on synthetic datasets serve as proxy measures of the uncertainty of our causal effect estimates. Estimating uncertainty is particularly challenging in networked data due to the violation of the i.i.d. assumption—interconnected nodes make traditional resampling or bootstrapping methods inappropriate. Hence, developing methods for uncertainty quantification in this context remains an important direction for future research.

Acknowledgements This work was funded in part by grants from the National Science Foundation (2226025), the National Center for Advancing Translational Sciences, and the National Institutes of Health (UL1 TR002014).

Author Contributions The original idea and foundation of this work were developed by A. Dalvi in close collaboration with V. Honavar, who provided critical insights and guidance throughout the conceptualization and evaluation phases via thorough discussions. The core code implementation of the algorithm was done by A. Dalvi, and N. Ashtekar verified its correctness. Both A. Dalvi and N. Ashtekar were involved in designing the data generation process. The benchmarking experiments were conducted by A. Dalvi, while N. Ashtekar generated the ablation plot results in Fig. 3 and verified the benchmarking results. A. Dalvi and N. Ashtekar prepared the initial draft of the manuscript. V. Honavar improved the clarity and technical depth of the paper by adding and removing content through multiple revision cycles. Finally, all three authors reviewed and approved the final version of the manuscript.

Data Availability Data and code for the C-HDNet experiments can be found in the following GitHub repository: <https://github.com/Abhishek-Dalvi410/C-HDNet>. This link is also mentioned in the manuscript. Parts of the benchmarking data are from the following paper <https://arxiv.org/pdf/1906.03485> (which is also cited in the manuscript).

Declarations

Conflict of interest The authors declare no Conflict of interest.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

References

- Chernozhukov V, Chetverikov D, Demirer M, Duflo E, Hansen C, Newey W, Robins J (2018) Double/debiased machine learning for treatment and structural parameters. Oxford University Press, Oxford
- Clivio O, Falck F, Lehmann B, Deligiannidis G, Holmes C (2022) Neural score matching for high-dimensional causal inference. In: International Conference on Artificial Intelligence and Statistics, PMLR, pp 7076–7110
- Chipman H, George E, McCulloch R (2010) BART: bayesian additive regression trees. *Ann Appl Stat* 4(1):266–298. <https://doi.org/10.1214/09-AOAS285>
- Chen H, Harinen T, Lee J-Y, Yung M, Zhao Z (2020) CausalML: python package for causal machine learning. arXiv preprint [arXiv:2002.11631](https://arxiv.org/abs/2002.11631)
- Chen D, Lin Y, Li W, Li P, Zhou J, Sun X (2020) Measuring and relieving the over-smoothing problem for graph neural networks from the topological view. In: Proceedings of the AAAI Conference on Artificial Intelligence 34:3438–3445
- Cox DR (1958) Planning of experiments. Wiley, New York
- Chen F, Wang Y-C, Wang B, Kuo C-CJ (2020) Graph representation learning: a survey. *APSIPA Trans Signal Inf Process* 9:15
- Dalvi A, Ashtekar N, Honavar VG (2024) Causal matching using random hyperplane tessellations. In *Causal Learning and Reasoning* (pp. 688–702). PMLR.
- Dirksen S, Mendelson S, Stollenwerk A (2022) Sharp estimates on random hyperplane tessellations. *SIAM J Math Data Sci* 4(4):1396–1419. <https://doi.org/10.1137/22M1485826>
- Guo R, Li Y, Li J, Candan KS, Raglin A, Liu H (2020) Ignite: a min-max game toward learning individual treatment effects from networked observational data. *IJCAI*
- Guo R, Li J, Liu H (2020) Counterfactual evaluation of treatment assignment functions with networked observational data. In: Proceedings of the 2020 SIAM International Conference on Data Mining, SIAM, pp 271–279
- Guo R, Li J, Liu H (2020) Learning individual causal effects from networked observational data. In: Proceedings of the 13th International Conference on Web Search and Data Mining, pp 232–240
- Hill J (2011) Bayesian nonparametric modeling for causal inference. *J Comput Graph Stat* 20(1):217–240. <https://doi.org/10.1198/jcgs.2010.08162>
- Hamilton W, Ying Z, Leskovec J (2017) Inductive representation learning on large graphs. In Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS'17). 1025–1035.
- Igelström E, Craig P, Lewsey J, Lynch J, Pearce A, Katikireddi SV (2022) Causal inference and effect estimation using observational data. *J Epidemiol Community Health* 76(11):960–966. <https://doi.org/10.1136/jech-2022-219267>
- Johansson FD, Shalit U, Sontag DA (2016) Learning representations for counterfactual inference. In: International Conference on Machine Learning
- Kanerva P (1992) Sparse distributed memory and related models. Technical report, NASA Ames Research Center
- Kanerva P (2009) Hyperdimensional computing: an introduction to computing in distributed representation with high-dimensional random vectors. *Cogn Comput* 1(2):139–159. <https://doi.org/10.1007/s12559-009-9009-8>
- Kingma DP (2013) Auto-encoding variational bayes. [arXiv:1312.6114](https://arxiv.org/abs/1312.6114)
- King G, Nielsen R (2019) Why propensity scores should not be used for matching. *Polit Anal* 27(4):435–454
- Kipf TN, Welling M (2017) Semi-supervised classification with graph convolutional networks. In: ICLR
- Kang J, Zhou M, Bhansali A, Xu W, Thomas A, Rosing T (2022) Relhd: a graph-based learning on feft with hyperdimensional computing. In: 2022 IEEE 40th International Conference on Computer Design (ICCD), pp 553–560. <https://doi.org/10.1109/ICCD56317.2022.00087>
- Louizos C, Shalit U, Mooij JM, Sontag D, Zemel R, Welling M (2017) Causal effect inference with deep latent-variable models. In Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS'17). 6449–6459.
- Neal B (2020) Introduction to causal inference from a machine learning perspective. Course Lect. Notes
- Neubert P, Schubert S, Protzel P (2019) An introduction to hyperdimensional computing for robotics. *KI - Künstliche Intelligenz* 33(4):319–330. <https://doi.org/10.1007/s13218-019-00623-z>
- Pearl J (2000) Causality: models, reasoning, and inference. Cambridge University Press, Cambridge
- Plan Y, Vershynin R (2014) Dimension reduction by random hyperplane tessellations. *Discrete Comput Geom* 51(2):438–461. <https://doi.org/10.1007/s00454-013-9561-6>
- Rosenbaum PR, Rubin DB (1983) The central role of the propensity score in observational studies for causal effects. *Biometrika* 70(1):41–55. <https://doi.org/10.1093/biomet/70.1.41>
- Rubin DB, Thomas N (1996) Matching using estimated propensity scores: relating theory to practice. *Biometrics* 35(4):417–446
- Rubin DB (1980) Bias reduction using mahalanobis-metric matching. *Biometrics* 36(2):293–298
- Rubin DB (2005) Causal inference using potential outcomes: design, modeling, decisions. *J Am Stat Assoc* 100(469):322–331
- Shi C, Blei D, Veitch V (2019) Adapting neural networks for the estimation of treatment effects. Proceedings of the 33rd International Conference on Neural Information Processing Systems. Article 225, 2507–2517.
- Shalit U, Johansson FD, Sontag D (2017) Estimating individual treatment effect: generalization bounds and algorithms. In: International Conference on Machine Learning, PMLR, pp 3076–3085
- Stuart EA (2010) Matching methods for causal inference: a review and a look forward. *Stat Sci Rev J Inst Math Stat* 25(1):1
- Thompson NC, Greenwald K, Lee K, Manso GF (2021) Deep learning's diminishing returns: the cost of improvement is becoming unsustainable. *IEEE Spectr* 58(10):50–55. <https://doi.org/10.1109/MSPEC.2021.9563954>
- Taie M, Kadry S (2017) Information diffusion in social networks, pp 165–184. https://doi.org/10.1007/978-3-319-53004-8_8
- Veličković P, Cucurull G, Casanova A, Romero A, Liò P, Bengio Y (2018) Graph attention networks. In: International Conference on Learning Representations
- Veitch V, Wang Y, Blei DM (2019) Using embeddings to correct for unobserved confounding in networks. [arXiv:1902.04114](https://arxiv.org/abs/1902.04114)

- Wager S, Athey S (2018) Estimation and inference of heterogeneous treatment effects using random forests. *J Am Stat Assoc* 113(523):1228–1242
- Wu Z, Pan S, Chen F, Long G, Zhang C, Philip SY (2020) A comprehensive survey on graph neural networks. *IEEE Trans Neural Netw Learn Syst* 32(1):4–24

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.