FeDCM: Federated Learning of Deep Causal Generative Models

Md Musfiqur Rahman[1] **Murat Kocaoglu**[1]

[1]School of Electrical and Computer Engineering, Purdue University,

Abstract

In many real-world settings, such as medicine and finance causal effect is a valuable metric for decision making. For many predictive tasks, causal mechanisms provide robust estimators while existing ML-driven predictors might be vulnerable to spurious correlations. In such settings, when data is decentralized and privacy must be preserved, federated learning plays an important role. However, causal inference in a federated learning setup is a largely unexplored research area. In this paper, we learn a proxy of the underlying structural causal model (SCM) with deep generative models from decentralized observational data sources possibly containing high-dimensional variables. Based on client preference or high dimensionality of variables, we modularize the SCM mechanisms and find the minimal subset appropriate for federated learning while having rest of the mechanisms trained on individual client's local data. When all connected together, the proxy SCM, named as the federated deep causal generative model (`FeDCM`), offers estimation of any identifiable causal effect. We perform extensive experiments to illustrate the utility and performance of our approach.

1 INTRODUCTION

There is growing appeal for causal inference in machine learning as causality can improve robustness, fairness, explainability, and data efficiency in machine learning systems [10, 49, 53, 29] . Predictive models $f : \mathcal{X} \rightarrow \mathcal{Y}$ are susceptible to spurious correlations due to unobserved confounding among variables and may produce biased and unfair predictions. Also, due to the dependence on domain-specific conditional distributions $P_1(y|x)$ (suppose domain 1), their predictions are not invariant and experience low test performance in a new domain (domain 2). Estimation of causal effects $P(y|do(x))$ or sampling of the corresponding interventional distribution alleviate these issues by removing any such confounding bias between X and Y [42, 19].

Many causal inference algorithms only work [15, 24] when the positivity (also known as overlap) assumption is satisfied, i.e., every joint combination of the variables values has positive probability. When we have small number of samples, it is unlikely that we will see all possible combinations in the data. As a result, some conditional distributions might be undefined (ex: P(recovery|age=90, sex=male, diseaseHistory=none)) Also, when we have low sample size, estimations of some conditional distributions might not be correct (sample mean $\neq$ population mean). If we plug in such biased conditional distributions in our causal effect estimators, the estimates will not be correct. Thus, the algorithms perform poor in low sample size settings. This problem is more prevalent when high-dimensional covariates such as images are present in the system. Some existing algorithms deal with such issues by learning from multiple datasets [45, 3, 9]. Even though combining multiple observational datasets resolve the data scarcity problem, in most real-world scenarios, such data are collected by different authorities (e.g., hospitals) and kept confidential at different locations. This creates a significant challenge.

Data scarcity is also a major problem in data analysis and current machine learning. Federated learning (FL) is an effective approach to train powerful ML models without different clients/parties having to share sensitive data. Client c trains the global model f_θ on its local datasets $D^c = \{x_i, y_i\}_{i=1}^{n_c}$ and share the model weights with the server so that when aggregated from multiple clients, f_θ learns to sample from the original distribution $P(y|x)$. Note that FL shares a global model weights across local clients whereas the existing causal inference algorithms require explicit probability tables for effect estimation. Even if we plan to share probability tables instead of model weights, such data statistics are infeasible to be computed for high-dimensional variables. For example, hospitals might share

Proceedings of the 41st Conference on Uncertainty in Artificial Intelligence (UAI 2025), PMLR 244:3507–3524.

probability distribution of getting pneumonia (n) given patient age (a), i.e, $P(n|a = 1)$. Such approach would not work if we aim to use X-ray images (x) for better prediction, i.e, $P(n|a, x)$. This invalidates the use of existing causal inference algorithms in federated learning setup.

A recent promising idea for estimating causal effect in the presence of high-dimensional variables (such as images) is to utilize neural networks [16, 35, 38, 22]. However, it is non-trivial to apply these approaches in the federated learning setup. Few works [11, 51, 52, 34, 47] have proposed to estimate specific treatment effects when data is decentralized across clients. These methods address only specific causal queries and are not suitable for arbitrary causal structures. Besides, for a new query, they have to redesign the training process to learn the conditional distributions and initiate the costly federated training from scratch.

In this paper, we assume that variables influencing each other in a system can be specified by causal mechanisms and modeled as a structural causal model (SCM). Mimicking the SCM of the environment allows us to measure the causal effect of *any* variable on others without being affected by spurious correlations, as long as the effect is structurally identifiable. Notably, learning all SCM mechanisms, rather than estimating a specific causal effect, better utilizes client connections during the federated communication phase. As a first step toward learning SCMs from decentralized data, we focus on existing approaches [18, 30, 50, 57, 36] that use deep generative models to learn the structural causal model from training data. A set of neural networks, called deep causal generative models (DCMs), are typically arranged according to the causal structure and trained on observational data. After convergence, they match the observational distribution and can be used to sample from identifiable interventional distributions or estimate causal effects.

New challenges appear when we aim to train a DCM in a federated fashion, as it requires training $|V|$ number of generative mechanisms of all V variables in the causal graph, i.e., $\{f_{V_i}\}_{V_i \in V}$ to match the joint distribution. A trivial implementation of a federated learning algorithm such as FedAVG [25] would be to consider $|V|$ global models which we share across clients and aggregate in the server. Such communication overhead is a major concern and infeasible for clients such as edge devices with limited compute and memory. A fundamental question is by how much we can reduce this complexity of distribution matching. Bayesian networks provide a way to modularly represent the joint distribution. Causal Bayesian Networks[1] enjoy the same compact representation (see Definition A.1). In the presence of latent variables, a term we call "modularity" allows the factorization of joint distribution into some products called c-factors. However, it is not clear how compactly we can represent the joint distribution and how much of it we can achieve in a federated learning setup. We do not want to transfer all causal mechanisms as global models between the server and the clients. Rather, we consider a set of models, $\mathcal{S}_{user}$ that clients prefer to share or based on high-dimensional variables that require federated learning. Our method `FeDCM` either accepts such a set or offers the minimal set of mechanisms $\mathcal{S}_{FL}$ containing $\mathcal{S}_{user}$ for federated training. We execute FL for the set of models, $\mathcal{S}_{FL}$ and utilize only local client data to learn the remaining mechanisms in the SCM.

To generalize this process, we solve two challenges: C1: How can we determine the minimal set of models $\mathcal{S}_{FL}$ that need to be trained collaboratively as FL global models based on $\mathcal{S}_{user}$: a client-proposed set or any high-dimensional variable set. C2: During and after FL training, how to utilize the global models $\mathcal{S}_{FL}$ to train the SCM mechanisms $\{f_{V_i}\}_{V_i \in V}$ consistently to match the joint distribution $P(v)$. Our proposed algorithm solves C1 by achieving what we call maximal modularity for training a causal model: we can learn mechanisms in each c-component (maximal subsets of nodes that are connected by unobserved confounding) independently. As part of solving C2, we note that, distributions corresponding to c-components (i.e., c-factors) are identifiable interventional distributions. Thus, instead of making $\mathcal{S}_{user}$, we make a proxy set of models $\mathbf{M}$ participate in FL and later use them to train $\mathcal{S}_{FL}$. Therefore, we train all SCM mechanisms on local observational data and simultaneously train $\mathcal{S}_{FL}$ on the interventional data generated by the proxy global models $\mathbf{M}$. After convergence, the DCM trained according to this approach represents the underlying true causal model. We can fix a specific value for a variable $X = x$ and perform ancestral sampling to generate interventional samples from $P(Z, Y|do(x))$ or estimate the causal effect. To the best of our knowledge, `FeDCM` is the first approach to efficiently learn *a neural SCM to answer any identifiable causal effect in a federated learning setup*. Precisely, our contributions are as follows:

- We propose a novel approach to learn a proxy structural causal model from multiple decentralized data sources containing both high and low-dimensional variables.
- We introduce the concept of maximal modularity that allows us to find the minimal set of SCM mechanisms for global training containing any client-preferred set of models and keep the communication cost low.
- We provide extensive empirical analysis on synthetic data showing the utility of our approach for learning a SCM in federated learning setup.

2 RELATED WORKS

In presence of data scarcity and privacy concerns, many existing works utilize deep learning based approach to deal

[1] A BN considers dependencies among a set of observed variables while a CBN considers causal dependencies among them.

with data heterogeneity, distribution shift [14, 2, 7, 56, 21, 43, 54] and spurious correlations [41, 48, 23]. Even though, these approaches are effective in their targeted problem instances, they might not generalize to any scenarios as they do not consider the causal relationships. There exists many causal inference-based approaches that learn causal structures or distributions from multiple datasets. Huang et al. [13] learns the causal graph from multiple data sets with non-identical variable sets. Tikka et al. [45] perform causal effect identification from multiple incomplete data sources while dealing with confounding and selection bias. Gresele et al. [9] utilize information from multiple datasets with overlapping variable set to obtain counterfactual inference. Bareinboim and Pearl [3] propose causal knowledge transportability by incorporating data from multiple causal domains. These methods assume access to all datasets and fail to adapt when data is decentralized.

Recently researchers have proposed different causal inference methods in the federated learning setup [4, 46, 33, 26, 17]. Ng and Zhang [28] estimates the Bayesian network structure from data that is partitioned across different parties with continuous optimization using the alternating direction method of multipliers. Ye et al. [55] propose federated learning of generalized linear causal networks from distributed datasets by simulating an annealing process and searching over the space of topological sorts. Gao et al. [8] employ federated learning to learn the underlying causal structure and the causal mechanisms from local heterogeneous data generated from additive noise models. Li et al. [20] propose an algorithm for structure and orientation learning utilizing summary statistics from distributed heterogeneous data.

Han et al. [11] utilizes semi-parametric density ratio weighting approach to provide treatment effect estimation where multiple clients contain heterogeneous covariate distributions. Xiong et al. [52] infer the average treatment effects by computing summary statistics locally using propensity scores and aggregating those across sites to obtain asymptotically normal point and variance estimators. Finally, Qiao et al. [34] estimate causal effect with data aggregated from multiple self-interested parties while rewarding them based on their unique statistical properties relating to a modified variant of the Shapley value. Vo et al. [47] divides the objective function into multiple components to estimate causal effects with federated training. These above works focus on estimation of specific queries and are not suitable for high-dimensional data in general.

3 PROBLEM DESCRIPTION

Definition 3.1 (Structural causal model (SCM) [31])**.** An SCM $\mathcal{M}$ is a 5-tuple $\mathcal{M} = (\mathcal{V}, \mathcal{N}, \mathcal{U}, \mathcal{F}, P(.))$, where each observed variable $V_i \in \mathcal{V}$ is realized as an evaluation of the function $f_i^* \in \mathcal{F}$ which looks at a subset of the remaining observed variables $Pa_i \subset \mathcal{V}$, an unobserved exogenous noise variable $E_i \in \mathcal{N}$, and an unobserved confounding (latent) variable $U_i \in \mathcal{U}$. This refers to the **semi-Markovian causal model**. $P(.)$ is a product joint distribution over all unobserved variables $\mathcal{N} \cup \mathcal{U}$.

Definition 3.2 (Acyclic Directed Mixed Graph (ADMG))**.** Each SCM induces a directed graph called the *causal graph*, or acyclic directed mixed graph (ADMG) with $\mathcal{V}$ as the vertex set. The directed edges are determined by which variables directly affect which other variable by appearing explicitly in that variable's function. Thus the causal graph is $G = (V, E)$ where $V_i \to V_j$ iff $V_i \in Pa_j$. The set Pa_j is called the parent set of V_j. We assume this directed graph is acyclic (DAG). Under the semi-Markovian assumption, each unobserved confounder can appear in the equation of exactly two observed variables. We represent the existence of an unobserved confounder $[U = U_X = U_Y] \in \mathcal{U}$ between X, Y in the SCM with a bidirected edge $X \leftrightarrow Y$ to the causal graph. These graphs are no longer DAGs although still acyclic. V_i is called an ancestor for V_j if there is a directed path from V_i to V_j. Then V_j is said to be a descendant of V_i. The set of ancestors of V_i in graph G is shown by $An_G(V_i)$. **C-components:** Given an ADMG G, a maximal subset of nodes where any two nodes are connected by bidirected paths is called a **c-component** $C(G)$. For any $S \in C(G)$, $P(S|\text{do}(\mathcal{V} \setminus S))$ is called a c-factor. We assume that we have access to the ADMG through some causal structure learning algorithm and expert knowledge.

Definition 3.3 (Causal effect and do-intervention)**.** A do-intervention $do(v_i)$ replaces the functional equation of V_i with $V_i = v_i$ without affecting other equations. The distribution induced on the observed variables after such an intervention is called an interventional distribution, shown by $P(\mathcal{V}|do(v_i))$. $P(\mathcal{V}|do(\emptyset)) = P(\mathcal{V})$ is called the observational distribution.

Definition 3.4 (Deep causal generative models (DCM) [18, 50, 36])**.** A neural net architecture $\mathbb{G}$ is called a deep causal generative model (DCM) for an ADMG $G = (\mathcal{V}, \mathcal{E})$ if it is composed of a collection of neural nets, one f_i (or interchangeably f_{V_i}) for each $V_i \in \mathcal{V}$ such that i) *each f_i accepts a sufficiently high-dimensional noise vector N_i,* ii) *the output of f_j is input to f_i iff $V_j \in Pa_G(V_i)$,* iii) $N_i = N_j$ *iff* $V_i \leftrightarrow V_j$. A DCM is trained to learn a proxy of the true SCM.

DCM generators are represented as $\mathbb{G} = \{f_1, ..., f_n\}$ parameterized by $\Theta = \{\theta_1, ..., \theta_{|\mathcal{V}|}\}$ where $n = |\mathcal{V}|$. Similar to the original data distribution, $P(\mathcal{V})$, we define $P_\theta(\mathcal{V})$ to be the distribution induced by the θ parameterized DCM. Noise vectors N_i replace both the exogenous noises and the unobserved confounders in the true SCM. They are of sufficiently high dimension to induce the observed distribution. We say that a DCM is *representative enough for an SCM* if the neural networks have sufficiently many parameters to induce the observed distribution induced by the SCM. For the

neural architectures of variables in the same c-component, we can consider conditional GANs [27], as they are effective in matching the joint distribution by feeding the same prior noise $N_i = N_j$ (as confounders) into multiple generators. Let $v = [v_1, v_2, ..., v_n]$ st $V_i \in \mathcal{V}$. $D(v)$ is real samples and $D(\hat{v}) \sim P_\theta(v)$ is DCM generated fake samples:

$$\hat{v}_i = f_i(\hat{pa}(V_i), u_{V_i}); f_i \in \{f_V : V \in \mathcal{V}\}, u_{V_i} \sim N(0, I)$$

The critic and generator (WGAN) loss functions are:

$$\begin{aligned} L_D &= \mathbb{E}_{v\sim P}[D(v)] - \mathbb{E}_{v\sim P_\theta}[D(v)] \\ L_G &= W(P, P^\theta) = -\mathbb{E}_{u\sim P(u)}[D(\hat{v})] \end{aligned} \tag{1}$$

The gradient updates are in such case are:

$$f_V^{(t+1)} = f_V^{(t)} - \eta \frac{\partial L_G}{\partial f_V}; \{f_V : V \in \mathcal{V}\} \tag{2}$$

With Defintion3.4, we have the following, similar to [50]:

Theorem 3.5. *[18, 50, 36] Consider any SCM $\mathcal{M} = (G, \mathcal{N}, \mathcal{U}, \mathcal{F}, P(.))$. A DCM $\mathbb{G} = \{f_1, ..., f_n\}$ for G entails the same identifiable interventional distributions as the SCM $\mathcal{M}$ if it entails the same observational distribution.*

Thus, even with high-dimensional variables, given a causal graph, in principle, any identifiable interventional query can be sampled from, with a DCM that fits the joint distribution.

Definition 3.6 (Interventional sampling with DCM)**.** Given the generators in a DCM, to perform a hard intervention $do(X = x)$ and produce samples accordingly, we manually set the values for the intervened variables as $X = x$ instead of using their neural network. Then, we feed forward those values into its children's mechanisms and execute **ancestral sampling** [5] to generate the rest of the variables.

Modular-DCM [36]: Modular-DCM use the c-factorization to modularize the DCM learning. Even though they point out the fact that each c-factor is an interventional distribution, they suggest learning a proxy distribution of each c-factor involving more variables than the c-component. For complicated graphs, the suggested proxy distribution might include all variables in the causal graph. This becomes wasteful especially in our considered federated learning setup. Here, we show that c-component based modularity is sufficient to learn the DCM and match the joint distribution $P(v)$ which has remained unexplored to date.

Federated Learning (FL): We consider a federated learning setting where C clients participate at each round of a training process coordinated at a central server. Training data is independent and identically distributed (iid) and is decentralized across multiple clients such that each client dataset D^c is sampled from a joint distribution $P(.)$. For a case of two variables, each sample in D^c is denoted as $(x, y) \in sup(X) \times sup(Y)$ with $sup(X), sup(Y)$ being the support of input X and output Y. Clients collaboratively train a mechanism: $F(w, x) : sup(X) \to sup(Y)$ to learn the conditional distribution $P(y|x)$. The global optimization problem is designed as the FedAVG algorithm [25]:

$$\min_w L(w) = \sum_{c=1}^{C} L_c(\theta) = \sum_{c=1}^{C} \mathbb{E}_{(x,y)\sim\mathcal{D}_c} l(F(w, x), y)$$

In the classic FedAvg [25] algorithm, the server samples a subset of C clients and broadcast the mechanism parameters w^t to those clients during round t. After performing local gradient updates, these clients return optimized mechanism w_c^t to the server. The server aggregates the local model to obtain a global model. In this paper, we consider a more general case where each client can share multiple models.

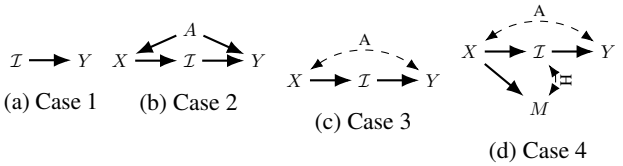
Problem setup: We assume that data sets for all clients are generated from the same underlying SCM. Dataset $\{D^c\}_{c=1}^C$ is collected from client c's environment with joint distribution $P(v)$ which is assumed to be generated from an unknown SCM $\mathcal{M}$. Our tasks are: i) to learn a DCM $\hat{\mathcal{M}}$ proxy to the true SCM $\mathcal{M}$ without exchanging any client data such that the observationa distribution $P(v)$ is matched, and ii) to estimate causal effects between any pair $X, Y \in \mathcal{V}$ or sample from the corresponding interventional distribution $P(y|do(x))$. Formally, in any client $c \in C$, our goal is to find a DCM $\hat{\mathcal{M}}$ s.t. $\arg\min_{\hat{\mathcal{M}}} d(P(v), P_\theta(\hat{v}))$. Since we do not want any spurious correlation in our prediction, our goal is always to obtain the interventional distributions (ex: $P(y|do(x))$ for arbitrary $X, Y \in \mathcal{V}$, and not the conditional distributions (ex: $P(y|x)$).

Assumptions: i) The observational dataset contains iid samples and distributed across multiple clients that collectively can represent the correct joint distribution $P(v)$ ii) we have access to the ADMG. iii) the causal model is semi-Markovian. iv) Each generator $f_i, \forall_i$ in the DCM can correctly learn the target distribution. (App A.3 for details).

4 METHODOLOGY

Suppose that due to privacy concerns or financial incentives, clients do not want to share any data and wish to share as few model weights as possible. Given a set of models proposed for federated training by the client $\mathcal{S}_{user}$, `FeDCM` evaluates if it is possible to learn an SCM by *only participating into the FL process with this set and use local data for training the remaining mechanisms*. If their intended mechanisms are not sufficient to learn the whole SCM, `FeDCM` rejects it and offers the minimal super set of $\mathcal{S}_{user}$ that are required to be learned collaboratively. For this purpose, we establish a maximal modularity concept considering the causal relations among the variables in the causal graph, more specifically the c-components.

Challenges of training a DCM in FL: Given the causal graph $G(\mathcal{V})$, we have $\{f_i\}_{i=1}^{|\mathcal{V}|}$ mechanisms in the causal



(a) Case 1 (b) Case 2 (c) Case 3 (d) Case 4

Figure 1: $\mathcal{I}$: X-ray image, Y: pneumonia prediction, X: symptoms, A: age. All clients maintain the same graph.

model. To learn $\{f_i\}; \forall i$ we are required to learn the true joint distribution $P(v)$ by training local models on client data and sharing necessary model weights. To be precise, consider the task of approximating the structural causal models in different setups shown in Fig. 1a- 1d. Fig. 1a shows the most common FL setup in which the goal is to predict pneumonia Y from X-ray images ($\mathcal{I}$). In Fig. 1b, we have access to patient symptoms X (w/ $X \to \mathcal{I}$) and patient age A. Older patients have a higher risk of developing different symptoms ($A \to X$) and are more likely to be diagnosed for pneumonia ($A \to Y$). In most hospital scenarios, the goal is to obtain pneumonia prediction (Y) in the form of causal effect estimation. Data are decentralized and suppose that clients proposed to share only the model weights of the mechanism $\mathcal{S}_{user} = \{f_Y\}$. Below, we show a few examples and then build a formal characterization of the set of mechanisms that need to be trained globally.

Trivial Solution: The trivial solution would be to always reject the client-proposed set and train the whole DCM: $\{f_i\}_{V_i \in \mathcal{V}}$ globally in a FL setup. That would ensure that $P(v)$ is matched. To obtain that, we need to minimize the following loss function (similar to Equation 1):

$$\min_{\theta} L(\theta) = \sum_{c=1}^{C} L_c(\theta) = \sum_{c=1}^{C} \sum_{v \in \mathcal{V}} \mathbb{E}_{[pa(v),v] \sim \mathcal{D}_c} l(\hat{v}, v)$$
$$\hat{v}_i = f_{\theta_i}(pa(V_i), u_{V_i}); u_{V_i} \sim N(0, I)$$

However, this will be computationally expensive and unnecessary. We aim to find the smallest set of mechanisms $\mathcal{S}_{FL}$ such that $f_{\mathcal{I}} \in \mathcal{S}_{FL}$ and we need to train them with global information to match $P(v)$.

Fig. 1a: Since, $P(y|do(i)) = P(y|i)$ for this graph, federated learning of only $f_y(i); \forall i \in \mathcal{I}$ is sufficient. Thus, we accept the client-proposed set and define $\mathcal{S}_{FL} = \{f_y\}$. We train the model $f_\theta \in \mathcal{S}_{FL}$ on local data D^c before sending it to the server so that, when aggregated from all clients, we get a global model for $P(y|x)$. In Fig. 1b, we can accept the client-proposed set and perform FL to learn the mechanism $f_Y(A, \mathcal{I})$ collaboratively and remaining mechanisms $f_X, f_A, f_{\mathcal{I}}$ locally. Thus, $\mathcal{S}_{FL} = \{f_y\}$.

In both Fig. 1c and 1d, patient age A is unobserved and represented as a bi-directed edge $X \leftrightarrow Y$. This forms the c-component $\{X, Y\}$. Note that in this graph $X \not\perp\!\!\!\perp Y|\mathcal{I}$ due to the unobserved confounder A. As suggested in the client-proposed set, if we train only f_Y independently with $\mathcal{I}$ as input, we have $y = f_Y(i, u_Y)$ which would create a wrong independence: $X \perp\!\!\!\perp Y|\mathcal{I}$. Here, X and Y share a joint $P(x, y)$ that must be matched to be consistent with the full joint $P(\mathcal{V})$. Training f_Y globally and f_X locally would not allow feeding the same noise U_A and matching $P(x, y)$. Rather, we need to train both $f_X(u_A)$ and $f_Y(u_A, i)$ together with the same confounding noise $U_A \sim \mathcal{N}(0, I)$. Thus, we reject $\mathcal{S}_{user}$ and offer the minimal set as $\mathcal{S}_{FL} = \{f_x, f_y\}$ that must participate in FL. Similarly, if $\mathcal{S}_{user} = \{\mathcal{I}\}$ in Fig. 1d, we have to reject it and propose $\mathcal{S}_{FL} = \{\mathcal{I}, M\}$ due to the unobserved confounder H.

Definition 4.1 (Maximal Modularity Set)**.** A set of mechanisms constructs an maximal modularity set denoted as $\mathcal{S}_{FL}$, if any mechanism $f \in \mathcal{S}_{FL}$ is trained independently with a loss function $L(f)$ while other mechanisms are trained with the loss function $L(\mathcal{S}_{FL} \setminus \{f\})$, they are not guaranteed to match the joint distribution $P(v)$.

4.1 MODULAR LEARNING OF DCM

Now, we characterize the maximal modularity set based on the causal relations among observed and unobserved variables in the given causal graph. Suppose, we have a causal graph containing observed variables and unobserved confounders as their parents (ADMG), that represents a semi-Markovian model and consisten with the observational distribution $P(v)$. Tian and Pearl [44] propose a method to factorize the joint distribution $P(v)$ into c-factors based on the c-component modules of the given ADMG.

$$P(v) = \prod_{s_i \in C(G)} P(s_i|\text{do}(pa(s_i)) \tag{3}$$

Here, $C(G)$ is the set of all c-components and $P(s_i|\text{do}(pa(s_i))$ is the c-factor corresponding to c-component S_i. Now, we establish a connection with c-factorization and training deep causal generative models (Definition 3.4: DCM). Note that in DCM we train the generative mechanism of all variables: $\mathbb{G} = \{f_V : V \in \mathcal{V}\}$ with a single loss function $L(\mathbb{G})$ (Equation 2) such that $P_\theta(\hat{v})$ matches the empirical joint distribution $P(v)$. The factorization in Equation 3 suggests that we can modularize the training process of mechanisms into c-components. If we can enforce our approximated SCM $\hat{\mathcal{M}}$ represented with the DCM $\mathbb{G}$ to match each of the c-factors, the joint distribution implied by the DCM will also match $P(v)$. More precisely, we can have $|C(G)|$ different loss functions $\{L_{S_i}(\{f_{V \in S_i}\}) : S_i \in C(G)\}$ and use them to independently train mechanisms in each c-component $\{f_V : V \in S_i\}$. The gradient updates are in such case are:

$$f_V^{(t+1)} = f_V^{(t)} - \eta \frac{\partial L_{S_i}}{\partial f_V}, \text{ for } V \in S_i, \text{ and } S_i \in C(G) \tag{4}$$

The main goal of this paper is to leverage federated learning to approximate the structural causal model. The above c-component based modularization allows us to train mechanisms in a c-component i.e., $\{f_V : V \in S_i\}$ together but

independently from other c-components $C(G) \setminus S_i$. Therefore, mechanisms in c-component S_i can utilize this opportunity to join collaborative federated training without affecting other mechanisms in the DCM. Thus, based on the client proposed set $\mathcal{S}_{user}$ we can find a partition of the c-components, i.e, $C(G) = C_l(G) \cup C_g(G)$ where $C_g(G)$ is the minimal set of c-components such that $\mathcal{S}_{user} \subseteq \{f_V : V \in C_g(G)\}$. We can re-write Equation 3 as,

$$P(v) = \prod_{s_i \in C_l(G)} P(s_i|\text{do}(pa(s_i)) \prod_{s_j \in C_g(G)} P(s_j|\text{do}(pa(s_j))$$

If $\mathcal{S}_{user} = \{f_V : V \in C_g(G)\}$ then we accept $\mathcal{S}_{user}$ as global models for FL. Otherwise, we offer $\mathcal{S}_{FL} = \{f_V : V \in C_g(G)\}$ as the minimal set for collaborative training and elect them as global models.

Now that we have selected the minimal subset of c-components as global models $\mathcal{S}_{FL}$, we update our training process. We train all mechanisms $\{f_V\}_{V \in \mathcal{V}}$ in the DCM jointly with a single loss function $L(\{f_V\}_{V \in \mathcal{V}})$ as usual (Equation 2) to match the observational distribution. Additionally, for the global mechanisms $f_V \in \mathcal{S}_{FL}$, we update their model weights with another loss function $L_{S_i}(\{f_{V \in S_i}\})$ aggregated with the original one. Let $v = [v_1, v_2, ..., v_m]$ such that $V_i \in C_g(G)$. $P^{in}_\theta(v)$ is the fake interventional distribution of DCM and $P^{inr}(v)$ is the **in**terventional **r**eal data distribution.

$$\begin{aligned} &\hat{v} \sim P^{in}_\theta(v); v \sim P^{inr}(v) \\ &\hat{v} = \{f_j(\hat{pa}(v_j), u_{V_j}\}; \forall f_j \in \{f_V : V \in C_g(G)\} \end{aligned} \tag{5}$$

The critic and generator loss functions ($L^{in}_D, L_{\mathcal{S}_{FL}}$ for matching interventional distributions are as follows:

$$\begin{aligned} L^{in}_D &= \mathbb{E}_{x \sim P^{inr}}[D^{in}(v)] - \mathbb{E}_{\hat{v} \sim P_\theta}[D^{in}(\hat{v})] \\ L^{in}_G &= W(P^{inr}, P^{in}_\theta) = -\mathbb{E}_{u \sim P(u)}[D(\hat{v})] \end{aligned} \tag{6}$$

And the gradient updates for DCM after training on both observational and interventional data are as follows:

$$f_V^{(t+1)} = \begin{cases} f_V^{(t)} - \eta \frac{\partial L_G}{\partial f_V}, & \text{if } V \notin \mathcal{S}_{FL} \\ f_V^{(t)} - \eta \frac{\partial (L_G + L^{in}_G)}{\partial f_V}, & \text{if } V \in \mathcal{S}_{FL} \end{cases} \tag{7}$$

This gives us an effective approach to learn a proxy of the SCM by training all mechanisms on local data while training the models in $\mathcal{S}_{FL}$ collaboratively with other clients. The loss function for DCM generators, $L_G = W(P^r, P_\theta)$ is optimized to match the local training distribution $P(v)$ and the additional loss function for generators $f_V \in \mathcal{S}_{FL}$ is $L^{in}_G = W(P^{inr}, P^{in}_\theta)$ is optimized to match global c-factor distribution $P(s_i|\text{do}(pa(s_i))$ where $S_i \subseteq C_g(G)$.

Note that, the c-factor is an interventional distribution and optimizing for the 2nd loss function L^{in}_G, would require real samples from that interventional distribution P^{inr}. However, we have only observational samples as training data which we use to optimize for the first loss function. How can we utilize observational samples to obtain samples from the c-factor interventional distribution and thus train the global models with them? In the next section, we provide a systematic approach as part of our novel `FeDCM` framework.

4.2 INTERVENTIONAL TRAINING DATA AND WHERE TO FIND THEM?

Given access to only observational data $D \sim P(v)$, we aim to minimize the loss function to train DCM mechanisms in each c-component $S_i \in C_g(G)$ utilizing federated learning. Equation 6 can be written as follow.

$$\begin{aligned} L^{in}_G &= W(P^{inr}, P^{in}_\theta) \\ &= W(P(S_i|do(pa(S_i))), P_\theta(S_i|do(pa(S_i)))) \end{aligned} \tag{8}$$

Equation 8 is a comparison between fake and real interventional distributions. To train DCM mechanisms $f \in \mathcal{S}_{FL}$ and learn this distributions, we need to compare its generated fake interventional samples, $\hat{D}^{in}$ against real interventional samples D^{in}. Even though we do not have access to any real interventional data, we implement the concept of causal effect identifiability to generate semi-synthetic interventional data and use them for our training.

Proposition 4.2 ([44, 40]). *If $C(G \setminus X) = \{S\}$ and $S \in C(G)$ then, $P_x(y) = \sum_{s \setminus y} \prod_{V_i \in S} P(v_i|v_\pi^{(i-1)})$*

This is a modification of c-factorization. Here, the condition implies that after removing the intervened variables from the original graph G, there exists a single c-component in the modified graph $G \setminus X$ that was also a c-component in the original graph G. This scenario occurs when there are no bi-directed edges from X to the c-component S. Then, we can utilize the above estimand for our causal query. In our case, for each query $P(S_i|do(pa(S_i))$, we consider the causal graph to be $G = Pa(S_i) \cup S_i$ and that satisfies the condition in Proposition 4.2. Thus, we can obtain:

$$P(s_i|do(pa(s_i)) = \prod_{\{j|V_j \in S_i\}} P(v_j|v_\pi^{(j-1)}) \tag{9}$$

Intuitively, identifiability gives us a way to express an identifiable interventional distributions, as a function of observational probability distributions. However, we can not train DCM generators with numeric values of probability tables rather need samples from the corresponding distribution.

Note that Equation 9 is a product of a set of conditional distributions in the form $P(v_j|v_\pi^{(j-1)})$. We follow [37] to generate samples from each such distribution by training a conditional model M_j that takes values of all ancestral dependent variables: $D[v_\pi^{(j-1)}]$ as input and generates v_j as outputs. We train $|S_i|$ number of conditional models $\mathbf{M} = \{M_1, ..., M_{|S_i|}\}$ with observational data each parameterized by $w = \{w_1, ..., w_{|S_i|}\}$. $P_w(v)$ is the distribution learned

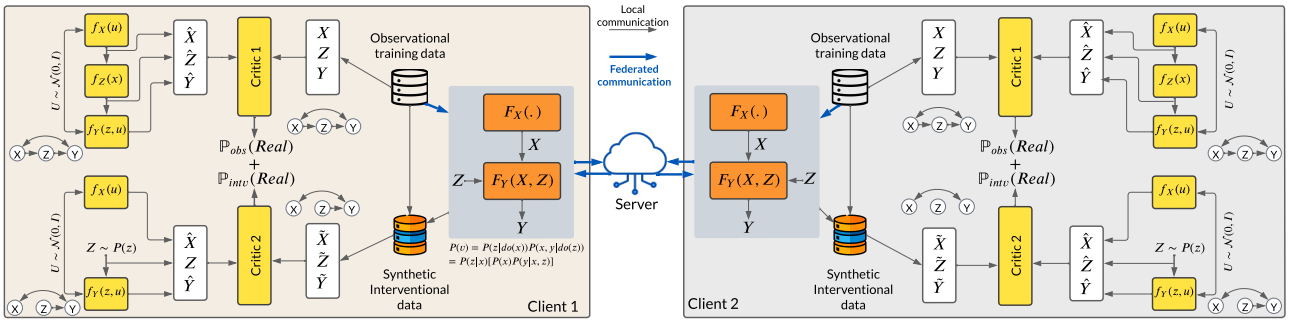


Figure 2: Algorithm simulation of FeDCM between two clients. Here $\mathcal{S}_{user} = \{f_x\}$ and $\mathcal{S}_{FL} = \{f_x, f_y\}$.

by $\mathbf{M}$. Here, $d(.,.)$ can be any loss function to measure distance between the distribution of two set of samples.

$$\begin{aligned} L_{FL} =& d(P(S|do(pa(S))), P_w(S|do(pa(S)))) \\ =& \sum_{i:V_i \in S} d(P(v_i|v_\pi^{(i-1)}), P_{w_i}(v_i|v_\pi^{(i-1)})) \end{aligned} \tag{10}$$

After convergence, we connect these $|S_i|$ trained models according to the conditional distributions in Equation 9.

$$\hat{v_j} = M_j(\hat{v}_\pi^{(j-1)}, pa(S_i)); [V_j, V_\pi^{j-1}] \subseteq S_i. \tag{11}$$

Finally, we feed output of one model as input to other models, i.e., perform ancestral sampling to generate samples from the interventional distribution $P(s_i|do(pa(s_i)))$.

Recall that, we elected $\mathcal{S}_{FL}$ as global models but they require interventional data for training. Therefore, we elect $\mathbf{M}$ for collaborative training as proxy to $\mathcal{S}_{FL}$ as we can train models in $\mathbf{M}$ on observational data using federated training. Later we can generate interventional data from $\mathbf{M}$ to train the models in $\mathcal{S}_{FL}$. As the proxy models are trained on observational data $D[Pa(S_i), S_i]$ only having access to models $M_{V_j}; V_j \in S_i, S_i \subset \mathcal{S}_{FL}$ and corresponding model weights are being shared during FL, privacy is preserved.

4.3 END-TO-END FEDCM FRAMEWORK

Here, we connect all pieces of our framework together. This is simulated in Fig. 2 for a frontdoor graph w/ $\mathcal{S}_{user} = \{X\}$.

Model initialization: All clients are assumed to agree upon a common causal graph, G. Each client initiates two sets of models. The first set (yellow) contains $\mathbb{G} = \{f_V\}_{V \in \mathcal{V}}$ to act as the local DCM (set of connected generators) according to G. Weights of each model f_i in the DCM are initialized as θ_i. The local DCM will generate fake observational data, $\hat{D}[\mathcal{V}] \sim P_\theta(\hat{v})$ (Fig 2: left-top) and fake interventional data, $\hat{D}[\mathcal{V}] \sim P_\theta(v|do(x)), X \subseteq \mathcal{V}$ (Fig 2: left-bottom).

We obtain all c-components $S_i \in C(G)$ from the causal graph. Based on the c-components in the causal graph, FeDCM either accepts the client-proposed set of models, $\mathcal{S}_{user}$ or proposes $\mathcal{S}_{FL}$: the c-component mechanisms of $C_g(G)$ that contains $\mathcal{S}_{user}$. Since corresponding c-factors are interventional distributions, we initiate a second set of models (orange) as proxy $\mathbf{M} = \{M_j\}_{V_j \in C_g(G)}$ to generate synthetic interventional data from c-factors, i.e., $\tilde{D}[Pa(S_i), S_i] \sim P_w(s_i|do(pa(s_i))$. Causal identifiability discussed in Section 4.2 ensures $\{M_j\}_{V_j \in C_g(G)}$ combinedly generate interventional data even though trained on only observational data. Each client performs the same process and join federated learning only for the second set of models.

Training global conditional models: We perform federated training for the set of models $\mathbf{M} = \{M_j\}_{V_j \in \mathcal{S}_{FL}}$ to learn conditional distributions collaboratively and utilize them to generate samples from the c-factor $P(s_i|do(pa(s_i))$. First, the model weights w_j of each function $M_j \in \mathbf{M}$ is updated as $w[V_j] \leftarrow w[V_j] - \eta\nabla\ell$ after training on local observational data. Next, each client sends the model weights to the server to be aggregated as the global model. The server receives the model weights from each client and takes a weighted average of the sent models. For the weights of each function M_j in $\mathbf{M}$, the server performs $w_{t+1}[V_j] \leftarrow \sum_{k=1}^{K} \frac{n_k}{n} w_{t+1}^k[V_j]$ at round t. Finally, the server broadcasts the new global models to each clients and they update their local models accordingly.

Training client local DCM architecture: The first training dataset $\tilde{D}^{in}[Pa(S_i), S_i]$ is the synthetic (real) interventional data for $P(s_i|do(pa(s_i))$ generated with conditional models in $\mathbf{M}$ and the second training dataset $D[\mathcal{V}]$ is the original observational training data. We train critic 1 to distinguish between DCM $\mathbb{G}$ generated fake $\hat{D}[\mathcal{V}]$ vs real observational data $D[\mathcal{V}]$ and critic 2 to distinguish between fake $\hat{D}[Pa(S_i), S_i]$ vs synthetic $\tilde{D}[Pa(S_i), S_i]$ interventional data. Now, instead of the loss function for c-factors in Equation 8, we now have:

$$\tilde{L}_G^{in} = d(P_w(S_i|do(pa(S_i))), P_\theta(S_i|do(pa(S_i)))) \tag{12}$$

According to triangle inequality, with the above loss we can upper bound the original DCM loss function in Equation 8:

$$\begin{aligned} & d(P(S_i|do(pa(S_i))), P_\theta(S|do(pa(S_i)))) \\ & \leq d(P(S_i|do(pa(S_i))), P_w(S|do(pa(S_i)))) \\ & + d(P_w(S_i|do(pa(S_i))), P_\theta(S_i|do(pa(S_i)))) \\ & \implies L_G^{in} \leq L_{FL} + \tilde{L}_G^{in} \end{aligned} \tag{13}$$

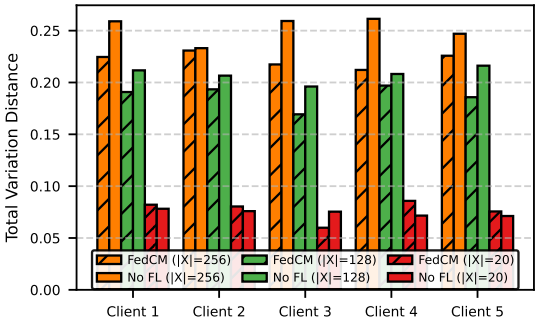


Figure 3: TVD of $P(X|Z)$ w/ 1000 training samples

Theorem 4.3. *Let A be any algorithm that, given a partition $\{S_1, S_2, \ldots, S_k\}$ of the nodes of a causal graph G, trains a deep causal generative model sequentially on $S_1, S_2, \ldots, S_k$ in that order to fit $P(S_i|S_1, S_2, \ldots S_{i-1})$, respectively.*

1. *If S_i are c-components of the graph G, then A is consistent, i.e., it fits the joint distribution correctly for any execution.*
2. *Conversely, when S_i are not c-components, then there exists a graph G for which algorithm A may fail for any order $S_1, S_2, \ldots S_k$, i.e., there exists a training execution that is inconsistent and algorithm A will not fit the joint distribution.*

5 EXPERIMENTS

We illustrate performance of `FeDCM` algorithm on synthetic and real-world IHDP data with extensive analysis. We provide additional experiments for non-identifiable causal effects in Appendix A.5. Our codes are made public.

Setup: We select the front-door causal graph in Figure 4a for our synthetic experiments. $\leftrightarrow$ implies that there is an unobserved confounder between Z and Y. We evaluate the performance of our algorithm on mainly two setups. First, we consider that each client has 1000 training samples and the training datasets have increasing support size of X, i.e., $X \in \{20, 128, 256\}$. For the second setup, we fix the support of X as $|X| = 20$ and evaluate our algorithm when each client has dataset size $|D^c| \in \{500, 1000, 1500, 2000\}$. Z and Y are considered to be binary. Suppose, clients wish to share only the model f_X, i.e., $\mathcal{S}_{user} = \{f_X\}$. Since X is a c-component itself, we accept it and perform federated training for f_X while training all f_Z, f_X, f_Y on local data. **Baseline (NoFL):** We consider a baseline where each client follows every step of our algorithm exactly but train on only local data and does not communicate with each other.

Varying support size of X: We illustrate the total variation distance (TVD) between the true distribution of $P(x|z)$ and the model approximated distribution for varying support size but for a fixed number of samples in Fig. 3. The orange, green and red bars represent X having support size equal 256, 128 and 20 respectively. The hatched bars represent `FeDCM` , and smooth bars represent the baseline NoFL. For client 1 that TVD for `FeDCM` (hatched bar) reduces from 0.22 to 0.19 to 0.08 as the support size of X reduces from 256 to 128 to 20, respectively. This behavior remains consistent for all clients (client 1-5). Note that `FeDCM` has lower TVD in (almost) all cases compared to NoFL (smooth bar). Since we keep the sample size fixed at 1000, when the support size is larger (ex: $|X| = 256$), the small number of samples is not sufficient to accurately learn $P(x|z)$ of a high-dimensional variable X. Thus, TVD loss is higher for $|X| = 256$ compared to $|X| = 20$. Our results show that federated training is highly effective when the mechanisms that participate in FL, i.e., $\mathcal{S}_{FL}$, are high dimensional.

(a) Query: $P(y|do(z))$. (b) Query: $P(y|do(t))$.

Figure 4: Causal graphs: synthetic and real (IHDP) datasets

Varying sample size: In figure 5, we show our performance on each client having sample sizes 500 (blue), 1000 (orange), 1500 (green) and 2000 (red) while keeping $|X| = 20$ as fixed. This plot represent how closely `FeDCM` approximated the true SCM as the TVD metric indicates the distance between the true joint distribution $P(v)$ and the distribution implied by the trained DCM $P_\theta(v)$. As the sample size increases to 500,1000, 1500 and 2000, the `FeDCM` TVD for client 1 (hatched) decreases to 0.13, 0.118, 0.105 and 0.094, accordingly. Here, `FeDCM` has a lower TVD in (almost) all cases compared to the baseline NoFL (smooth bar) even for X having low dimension. Figure 5 shows that with small support size (i.e., 20), when clients have high number of samples (eg., red, 2000), FL does not significantly improve the training performance as each client has enough samples to obtain a good estimator for $P(x|z)$. Thus, both individual clients (smooth red bars) and `FeDCM` (hatched red bars) obtain similar performance. However, if we have small number of samples (eg., blue, 500), clients can not learn the unbiased estimator for the distribution by themselves in individual training. Our method exploits the federated learning setup to obtain a better estimator of $P(x|z)$ compared to individual clients. Thus when the sample size decreases (small sample size), even for small support sizes, the performance gap between federated training (hatched blue bars) and individual training (smooth blue bars) increases. After matching the joint distribution with low TVD, prediction for identifiable causal effect should be close to the ground truth.

5.1 EXPERIMENTAL ANALYSIS ON IHDP

Dataset: We performed an experiment on a real benchmark dataset, the Infant Health and Development Program

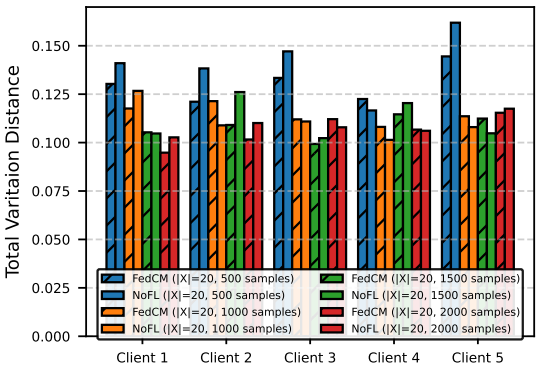


Figure 5: TVD of $P(Z, X, Y)$ w/ different sample sizes.

(IHDP) dataset [12]. It contains 747 records with a total of 27 variables of varying data types that increase problem complexity. It has i) 25 covariates: 6 continuous variables and 19 discrete variables, ii) 1 treatment variable: discrete, iii) 1 outcome variable: continuous. To mimic the federated setup, samples are randomly distributed to the clients. Even though non-iid distribution is not considered here, due to the small number of samples in each client, data heterogeneity will arise. To compare with our baselines, we aim to estimate the Average Treatment Effect (ATE): $ATE = \mathbb{E}[P(Y \mid do(T = 1))] - \mathbb{E}[P(Y \mid do(T = 0))]$ Note that, we assume the causal graph in Figure 4b for the IHDP dataset. Our main baseline, Vo et al. (2022), assumes that they do not have access to the original set of covariates. Rather, they use some proxy (X) of them. They try to obtain a posterior of the covariates Z from the proxy X, treatment T, and outcome Y. In contrast, we consider a one-to-one invertible mapping between covariates and the proxy, i.e., $Z = X$. Hence, we assume full observability of Z and no unobserved confounders.

Solution Design: We aim to learn a deep causal model consistent with our assumed causal graph: a proxy of the true SCM. Ideally, we train G_Z, G_T, G_Y to match $P(Z, T, Y)$. However, for our target ATE (same as the baselines), we need to estimate $P(Y \mid do(T = 1))$ and $P(Y \mid do(T = 0))$. By the backdoor criterion: $P(Y \mid do(T)) = \sum_z P(Y \mid z, T)P(z)$. Two approaches can be used to estimate the causal effects. i) Approach 1: Federated learning (FL) trains model M_Z to learn $P(z)$ and model M_Y to learn $P(y \mid z, t)$. Then, generate $Z \sim P(Z)$ using M_Z, and sample Y from M_Y using $P(y \mid z, t)$. These Y samples approximate $P(Y \mid do(T))$. ii) Approach 2: FL trains only M_Y to learn $P(y \mid z, t)$. Use Z directly from the training dataset ($Z \sim P(Z)$) and feed into M_Y to sample $Y \sim P(y \mid z, t)$. In both cases, the generated Y are used to estimate ATE. We ran our algorithm for 200 federated learning rounds, with each client trained for 2000 epochs per round. We utilize a GAN architecture to learn the joint distributions. To verify our estimation, we need a reference value for ATE. As no ground truth exists, we used: i) Vo et al. [47]'s implementation: ground truth ATE ≈ 3.98 ii) Dowhy package [39] (propensity score matching on full dataset): also yielded similar value. Thus, we consider 3.98 as our reference ATE.

Method vs ATE error (Reference ATE = 3.98)			
BART_ag	1.3 ± 0.05	TARNet_ag	2.5 ± 0.06
X-Learner_ag	1.2 ± 0.09	CFR-wass_ag	2.7 ± 0.05
R-Learner_ag	1.0 ± 0.07	CFR-mmd_ag	2.5 ± 0.03
OthoRF_ag	1.3 ± 0.09	CEVAE_ag	2.1 ± 0.09
FedCI	0.5 ± 0.09	CausalRFF	0.5 ± 0.16
Ours_3clients_w/50samples		$\mathbf{0.418 \pm 0.24}$	
Ours_12clients_w/50samples		$\mathbf{0.351 \pm 0.22}$	
Ours_3clients_w/200samples		$\mathbf{0.271 \pm 0.16}$	

Baselines Comparison: We follow the same experimental setup as described by Vo et al. [47], and compare our performance with the results reported in their paper, obtained by their method and other baselines. We consider the last 20 global rounds of our model training for evaluation. Based on the reference and predicted ATE, we calculate the mean and standard deviation of the ATE error for each client and report the average across all clients. According to Vo et al. [47], these baselines use 50, 100, and 99 samples for train, test, and validation sets per client, with number of clients = 3. To illustrate the scale of our experiment, we consider 3 setups: i) 3 clients each with 50 samples, ii) 12 clients each with 50 samples, iii) 3 clients each with 200 samples.

We have two main observations. **Observation 1:** In all setups (i, ii, iii), we obtain relatively small mean ATE error compared to other baselines. **Observation 2:** Sorting ATE errors yields: a: 3 clients 50 samples $>$ b: 12 clients 50 samples $>$ c: 3 clients 200 samples. a $>$ b shows that our method can benefit from federated learning to reduce ATE error. b $>$ c shows how sample size impacts error in FL settings. The observed standard deviation is likely due to GAN training on small datasets. Moreover, for the model that matches $P(Z)$ we observe max TVD loss $= 0.0768$ (discrete covariates). For the model that matches $P(Y \mid Z, T)$, we observe Wasserstein distance $= 0.0147$ (continuous Y). These metrics indicate reasonable convergence of the models.

6 CONCLUSION

We explore the federated learning to learn a proxy of the structural causal model from distributed datasets. We introduce maximal modularity and propose a framework where a set of models trains to learn the SCM from local data while another set of models participates in the collaborative training to aid global information to the local models. After convergence, the DCM can estimate any identifiable causal effects. In our future work, we aim to remove the assumption on having access to the true causal graph.

Acknowledgements

This research has been supported in part by NSF CAREER 2239375, IIS 2348717, Amazon Research Award and Adobe Research. We also thank Ilya Shpitser for sharing helpful insights on the identifiability of c-factors.

References

[1] Vahid Balazadeh Meresht, Vasilis Syrgkanis, and Rahul G Krishnan. Partial identification of treatment effects with implicit generative models. *Advances in Neural Information Processing Systems*, 35:22816–22829, 2022.

[2] Wenxuan Bao, Tianxin Wei, Haohan Wang, and Jingrui He. Adaptive test-time personalization for federated learning. *Advances in Neural Information Processing Systems*, 36, 2024.

[3] Elias Bareinboim and Judea Pearl. A general algorithm for deciding transportability of experimental results. *Journal of causal Inference*, 1(1):107–134, 2013.

[4] Rohit Bhattacharya, Tushar Nagarajan, Daniel Malinsky, and Ilya Shpitser. Differentiable causal discovery under unmeasured confounding. In *International Conference on Artificial Intelligence and Statistics*, pages 2314–2322. PMLR, 2021.

[5] Christopher M Bishop. Pattern recognition and machine learning. *Springer google schola*, 2:645–678, 2006.

[6] Patrick Chao, Patrick Blöbaum, and Shiva Prasad Kasiviswanathan. Interventional and counterfactual inference with diffusion models, 2023.

[7] Junbao Chen, Jingfeng Xue, Yong Wang, Zhenyan Liu, and Lu Huang. Classifier clustering and feature alignment for federated learning under distributed concept drift. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=6ejpSVIiIl.

[8] Erdun Gao, Junjia Chen, Li Shen, Tongliang Liu, Mingming Gong, and Howard Bondell. Federated causal discovery. 2021.

[9] Luigi Gresele, Julius Von Kügelgen, Jonas Kübler, Elke Kirschbaum, Bernhard Schölkopf, and Dominik Janzing. Causal inference through the structural causal marginal problem. In *International conference on machine learning*, pages 7793–7824. PMLR, 2022.

[10] Limor Gultchin. *Casual and trustworthy machine learning: methods and applications*. PhD thesis, University of Oxford, 2023.

[11] Larry Han, Jue Hou, Kelly Cho, Rui Duan, and Tianxi Cai. Federated adaptive causal estimation (face) of target treatment effects. *arXiv preprint arXiv:2112.09313*, 2021.

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

[13] Biwei Huang, Kun Zhang, Mingming Gong, and Clark Glymour. Causal discovery from multiple data sets with non-identical variable sets. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 10153–10161, 2020.

[14] Wenke Huang, Mang Ye, Zekun Shi, He Li, and Bo Du. Rethinking federated learning with domain shift: A prototype view. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 16312–16322. IEEE, 2023.

[15] Inwoo Hwang, Yesong Choe, Yeahoon Kwon, and Sanghack Lee. On positivity condition for causal inference. In *Forty-first International Conference on Machine Learning*, 2024.

[16] Connor T Jerzak, Fredrik Johansson, and Adel Daoud. Image-based treatment effect heterogeneity. *arXiv preprint arXiv:2206.06417*, 2022.

[17] Rémi Khellaf, Aurélien Bellet, and Julie Josse. Federated causal inference: Multi-centric ATE estimation beyond meta-analysis. In *The 28th International Conference on Artificial Intelligence and Statistics*, 2025. URL https://openreview.net/forum?id=1QovyEffl5.

[18] Murat Kocaoglu, Christopher Snyder, Alexandros G Dimakis, and Sriram Vishwanath. Causalgan: Learning causal implicit generative models with adversarial training. In *International Conference on Learning Representations*, 2018.

[19] Kenneth Lee, Md Musfiqur Rahman, and Murat Kocaoglu. Finding invariant predictors efficiently via causal structure. In *Uncertainty in Artificial Intelligence*, pages 1196–1206. PMLR, 2023.

[20] Loka Li, Ignavier Ng, Gongxu Luo, Biwei Huang, Guangyi Chen, Tongliang Liu, Bin Gu, and Kun Zhang. Federated causal discovery from heterogeneous data. *arXiv preprint arXiv:2402.13241*, 2024.

[21] Xinting Liao, Weiming Liu, Pengyang Zhou, Fengyuan Yu, Jiahe Xu, Jun Wang, Wenjie Wang, Chaochao Chen, and Xiaolin Zheng. FOOGD: Federated collaboration for both out-of-distribution generalization and detection. In *The Thirty-eighth Annual*

Conference on Neural Information Processing Systems, 2024. URL https://openreview.net/forum?id=D6MQrw9HFu.

[22] Christos Louizos, Uri Shalit, Joris M Mooij, David Sontag, Richard Zemel, and Max Welling. Causal effect inference with deep latent-variable models. *Advances in neural information processing systems*, 30, 2017.

[23] Shuran Ma, Weiying Xie, Daixun Li, Haowei Li, and Yunsong Li. Reducing spurious correlation for federated domain generalization. *arXiv preprint arXiv:2407.19174*, 2024.

[24] Andreas Markoulidakis, Peter Holmans, Philip Pallmann, Monica Busse, and Beth Ann Griffin. How balance and sample size impact bias in the estimation of causal treatment effects: a simulation study. *arXiv preprint arXiv:2107.09009*, 2021.

[25] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pages 1273–1282. PMLR, 2017.

[26] Osman Mian, David Kaltenpoth, Michael Kamp, and Jilles Vreeken. Nothing but regrets—privacy-preserving federated causal discovery. In *International Conference on Artificial Intelligence and Statistics*, pages 8263–8278. PMLR, 2023.

[27] Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.

[28] Ignavier Ng and Kun Zhang. Towards federated bayesian network structure learning with continuous optimization. In *International Conference on Artificial Intelligence and Statistics*, pages 8095–8111. PMLR, 2022.

[29] Alvaro PARAFITA and Jordi VITRIA. Deep causal graphs for causal inference, black-box explainability and fairness. In *Artificial Intelligence Research and Development: Proceedings of the 23rd International Conference of the Catalan Association for Artificial Intelligence*, volume 339, page 415. IOS Press, 2021.

[30] Nick Pawlowski, Daniel Coelho de Castro, and Ben Glocker. Deep structural causal models for tractable counterfactual inference. *Advances in Neural Information Processing Systems*, 33:857–869, 2020.

[31] J. Pearl. *Causality*. Cambridge University Press, 2009. ISBN 9781139643986. URL https://books.google.com/books?id=LLkhAwAAQBAJ.

[32] Judea Pearl. From bayesian networks to causal networks. In *Mathematical models for handling partial knowledge in artificial intelligence*, pages 157–182. Springer, 1995.

[33] Jose M Peña. Learning acyclic directed mixed graphs from observations and interventions. In *Conference on Probabilistic Graphical Models*, pages 392–402. PMLR, 2016.

[34] Rui Qiao, Xinyi Xu, and Bryan Kian Hsiang Low. Collaborative causal inference with fair incentives. 2023.

[35] Wei Qin, Hanwang Zhang, Richang Hong, Ee-Peng Lim, and Qianru Sun. Causal interventional training for image recognition. *IEEE Transactions on Multimedia*, 2021.

[36] Md Musfiqur Rahman and Murat Kocaoglu. Modular learning of deep causal generative models for high-dimensional causal inference. In *Forty-first International Conference on Machine Learning*, 2024. URL https://openreview.net/forum?id=bOhzU7NpTB.

[37] Md Musfiqur Rahman, Matt Jordan, and Murat Kocaoglu. Conditional generative models are sufficient to sample from any causal effect estimand. *arXiv preprint arXiv:2402.07419*, 2024.

[38] Uri Shalit, Fredrik D Johansson, and David Sontag. Estimating individual treatment effect: generalization bounds and algorithms. In *International conference on machine learning*, pages 3076–3085. PMLR, 2017.

[39] Amit Sharma and Emre Kiciman. Dowhy: An end-to-end library for causal inference. *arXiv preprint arXiv:2011.04216*, 2020.

[40] Ilya Shpitser and Judea Pearl. Complete identification methods for the causal hierarchy. 2008.

[41] Chandan Singh, Armin Askari, Rich Caruana, and Jianfeng Gao. Augmenting interpretable models with large language models during training. *Nature Communications*, 14(1):7913, 2023.

[42] Adarsh Subbaswamy, Peter Schulam, and Suchi Saria. Preventing failures due to dataset shift: Learning predictive models that transport. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 3118–3127. PMLR, 2019.

[43] Yue Tan, Chen Chen, Weiming Zhuang, Xin Dong, Lingjuan Lyu, and Guodong Long. Is heterogeneity notorious? taming heterogeneity to handle test-time shift in federated learning. *Advances in Neural Information Processing Systems*, 36, 2024.

[44] Jin Tian and Judea Pearl. *A general identification condition for causal effects*. eScholarship, University of California, 2002.

[45] Santtu Tikka, Antti Hyttinen, and Juha Karvanen. Causal effect identification from multiple incomplete data sources: A general search-based approach. *arXiv preprint arXiv:1902.01073*, 2019.

[46] Steven Turnbull. *Constraint-Score Hybrid Structure Learning for Directed Acyclic Graphs With Latent Confounders*. University of California, Los Angeles, 2024.

[47] Thanh Vinh Vo, Arnab Bhattacharyya, Young Lee, and Tze-Yun Leong. An adaptive kernel approach to federated learning of heterogeneous causal effects. *Advances in Neural Information Processing Systems*, 35:24459–24473, 2022.

[48] Xiaoyang Wang, Han Zhao, Klara Nahrstedt, and Sanmi Koyejo. Personalized federated learning with spurious features: An adversarial approach. *Transactions on Machine Learning Research*, 2024.

[49] Yongkai Wu, Lu Zhang, Xintao Wu, and Hanghang Tong. Pc-fairness: A unified framework for measuring causality-based fairness. *Advances in neural information processing systems*, 32, 2019.

[50] Kevin Xia, Kai-Zhan Lee, Yoshua Bengio, and Elias Bareinboim. The causal-neural connection: Expressiveness, learnability, and inference. *Advances in Neural Information Processing Systems*, 34:10823–10836, 2021.

[51] Ruoxuan Xiong, Allison Koenecke, Michael Powell, Zhu Shen, Joshua T Vogelstein, and Susan Athey. Federated causal inference in heterogeneous observational data. *arXiv preprint arXiv:2107.11732*, 2021.

[52] Ruoxuan Xiong, Allison Koenecke, Michael Powell, Zhu Shen, Joshua T Vogelstein, and Susan Athey. Federated causal inference in heterogeneous observational data. *Statistics in Medicine*, 42(24):4418–4439, 2023.

[53] Depeng Xu, Yongkai Wu, Shuhan Yuan, Lu Zhang, and Xintao Wu. Achieving causal fairness through generative adversarial networks. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence*, 2019.

[54] Yiyuan Yang, Guodong Long, Tao Shen, Jing Jiang, and Michael Blumenstein. Dual-personalizing adapter for federated foundation models. *arXiv preprint arXiv:2403.19211*, 2024.

[55] Qiaoling Ye, Arash A Amini, and Qing Zhou. Federated learning of generalized linear causal networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024.

[56] Shuyang Yu, Junyuan Hong, Haotao Wang, Zhangyang Wang, and Jiayu Zhou. Turning the curse of heterogeneity in federated learning into a blessing for out-of-distribution detection. In *2023 International Conference on Learning Representations*, 2023.

[57] Weijia Zhang, Lin Liu, and Jiuyong Li. Treatment effect estimation with disentangled latent factors. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 10923–10930, 2021.

FeDCM: Federated Learning of Deep Causal Generative Models (Supplementary Material)

Md Musfiqur Rahman[1] **Murat Kocaoglu**[1]

[1]School of Electrical and Computer Engineering, Purdue University,

A ADDITIONAL DETAILS

Definition A.1 (Causal Bayesian Network [32])**.** A Bayesian network considers dependencies among a set of observed variables while a causal Bayesian network considers causal dependencies among them. When we perform an intervention on a variable (i.e. set a variable to a fixed value), it affects the variables that have causal relations with it and keeps remaining variables unaffected. A Bayesian network does not have the concept of intervention as it considers only correlation and not causal relations. However, suppose the Bayesian network has a consistent structure such that i) after performing an intervention, no other nodes except the descendants are affected due to the intervention ii) and this is true for all possible interventions, we can call it a causal Bayesian network.

Since our goal is causal effect estimation, the reviewer is correct to observe that our focus is on causal Bayesian networks in this paper.

A.1 FEDCM MAIN CONTRIBUTIONS

Our contributions in this paper are two folds: First (algorithmic improvement with maximum modularity): our maximum modularity result shows that to learn a c-factor distribution, we do not need access to variables outside the c-component unlike [36] who construct a complicated structure called, H-graph and learn a joint distribution larger than the c-factor in many scenarios. Training a smaller number of models is particularly helpful in a federated setting to reduce communication cost.

Secondly (Adaptation in federated learning setting): Although there are some recent works [30, 36, 50, 6] that learn deep/neural causal models from observational data, all of them assume access to the entire dataset, i.e., they are designed for centralized settings. We propose the first approach to learn a deep causal model in the federated learning setup.

A.2 DATA TYPES:

Note that in our synthetic experiments, all of our variables are discrete (with finite support size). However, this is a specific instance of experiment and not a limitation of our approach. Any variable (covariates, mediators, outcome variables), except the intervened variable can be discrete, continuous or high-dimensional (ex: images) and our method will work when employed with appropriate generative models. However, when the intervened variable (treatment) is continuous, causal effect estimation is generally non-regular and requires a more careful handling [1].

A.3 VIOLATION OF ASSUMPTIONS AND FUTURE WORKS

Below we provide some details on our assumptions and what challenges one might face if assumptions are violated or relaxed.

A.3.1 Assumption of having access to the causal graph

The access to ADMG assumption can be relaxed by employing any federated-learning-compatible causal discovery algorithm, as a pre-processing step. We aim to address the challenges associated with non-iid data in our future works.

A.3.2 What if some clients only observe a subset of variables?

If clients observe variable values located at different parts of the causal graph, it might be challenging to obtain the whole joint distribution. But as our method offers c-component based modular training, clients do not need to observe the whole joint. Additionally, we can resort to approaches such as [9] which merges available causal marginal information from given observations of subsets of variables in a causal graph. This is an interesting direction we aim to explore in our future works.

A.3.3 The impact of model mis-specification on performance

One interesting future direction is to see how the estimation changes when we have the incorrect causal graph. We might assume incorrectly i) presence of an edge, ii) orientation of an edge, iii) unobserved confounders etc. If iii) happens, i.e., we assume that we have observed all confounders but there still exists some unobserved confounder, then the causal effect will be non-identifiable and we can obtain a bound using maximization and minimization in the NCM/DCM algorithm.

For i) and ii) one solution might be to iterate all possible choice of edge presence/orientation to obtain the set of causal graphs and then train a DCM for those. This will give us a set of possible causal effects. In our future work, we plan to explore how these possibilities will work in a federated learning setup.

A.4 COMPLEXITY ANALYSIS

Communication complexity: The communication cost for a single client is $O(T * |C_{max}(G)| * m)$ where T : global rounds, m : the communication cost for sending all weights of a single neural network and $C_{max}(G)$: the largest c-component in the graph.

Below we discuss in detail.

Suppose, clients propose a set of mechanisms, S_{user} for collaborative learning. If corresponding variables of S_{user} does not form a whole c-component in the graph, we reject the set and offer a super-set S_{FL} that construct a c-component in the graph. Note: A c-component is maximal subsets of nodes that are connected by unobserved confounders.

Let S_{FL} correspond to the largest c-component in the graph, $C_{max}(G)$. This implies that we have to share model weights for $|C_{max}(G)|$ number of neural networks (NN) to participate in federated training. If m is the communication cost for sending all weights of a single neural network, our communication cost for a client is= $O(|C_{max}(G)| * m)$.

If there are total T global rounds, the communication cost for a single client is $O(T * |C_{max}(G)| * m)$.

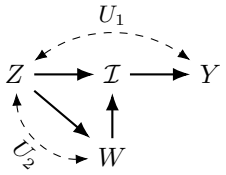
Server computational complexity: As the server will aggregate weights of $|C_{max}(G)|$ different models, its computation time complexity is $O(|C_{max}(G)|)$.

Client computational complexity: To join the federated learning, each client will have to train $|C_{max}(G)|$ number of NN models. Next, clients will learn the deep causal model with local data and interventional data generated from these $|C_{max}(G)|$ NN models.

Suppose the set of all variables in the SCM is $\mathcal{V}$. As each client learns a deep causal model $\mathbb{G}$ that consists of $|\mathcal{V}|$ NN models, it would require $O(|\mathcal{V}|)$ time to train them. So, the total complexity of NN model training would be $O(|C_{max}(G)| + |\mathcal{V}|)$.

A.5 ADDITIONAL EXPERIMENTS

We have shared our codes at: github.com/musfiqshohan/fedcm



The unobserved variables $\{U_1, U_2\}$ have a support size of 3, while the observed variables $\{Z, W, \mathcal{I}, Y\}$ have a support size of 2. In this graph, the unobserved confounder U_2 makes the causal effect $P(Y \mid do(Z))$ *non-identifiable*. The goal is to obtain a bound over all possible causal effects such that the true effect lies within it. The challenge is that data is decentralized and user preferred model that is allowed to train globally is $\mathcal{S}_{user} = \{\mathcal{I}\}$.

We employ **FeDCM** to design our solution. Here we have two c-components $\{Z, W, Y\}$ and $\{\mathcal{I}\}$. We train a conditional model M to learn $P(\mathcal{I} \mid Z, W)$ in a federated manner, as it belongs to a different c-component. Next, we train the deep causal model: $\mathbb{G}_Z, \mathbb{G}_W, \mathbb{G}_Y, \mathbb{G}_\mathcal{I}$ such that:

i) the observational distribution $P(z, w, i, y)$ of local data is matched,

ii) the interventional distribution $P(\mathcal{I} \mid do(z))$ is matched using samples generated from the globally trained model M, and

iii) our target query $P(y = 1 \mid do(z = 1))$ is maximized (or minimized).

We use the total variation distance (TVD) to assess how well the distributions are learned. In practice, TVD is never exactly zero due to approximation errors during model training. Therefore, we construct bounds for the causal effect whenever the corresponding joint distribution is matched with TVD < 0.15. Below, we provide the bounds for three clients:

TVD < 0.15	**Ground Truth** $P(y = 1 \mid do(z = 1))$	**Client 1**	**Client 2**	**Client 3**
		Bound [min, max]	Bound [min, max]	Bound [min, max]
$\lvert X\rvert = 2$	0.186	[0.125, 0.2692]	[0.0674, 0.2998]	[0.0572, 0.2728]

Table 1: Bounds on $P(y = 1 \mid do(z = 1))$ estimated by FeDCM for three clients.

B THEORETICAL ANALYSIS

Theorem B.1. *Let A be any algorithm that, given a partition $\{S_1, S_2, \ldots, S_k\}$ of the nodes of a causal graph G, trains a deep causal generative model sequentially on $S_1, S_2, \ldots, S_k$ in that order to fit $P(S_i|S_1, S_2, \ldots S_{i-1})$, respectively.*

1. *If S_i are c-components of the graph G, then A is consistent, i.e., it fits the joint distribution correctly for any execution.*
2. *Conversely, when S_i are not c-components, then there exists a graph G for which algorithm A may fail for any order $S_1, S_2, \ldots S_k$, i.e., there exists a training execution that is inconsistent and algorithm A will not fit the joint distribution.*

Proof. Let us define the bi-directed neighbors, $bNb(X)$ as the set of variables such that $\forall V \in bnb(X)$ there exists a bi-directed edge $V \leftrightarrow X \in E$.

(1.) Consider a c-component $S_i \in C(G)$ and $S_i = \{X_1, ..., X_j, X_{j+1}, ..., X_m\}$. Let variable X_j and X_{j+1} have a bi directed edge between them, i.e, $X_j \leftrightarrow X_{j+1}$ and the shared common confounder is U. Since they share a common confounder, $X_j \not\perp\!\!\!\perp X_{j+1}$. Thus we need to feed a same confounding (Gaussian) noise $U \sim N(0, I)$ as input to both f_{X_j} and $f_{X_{j+1}}$ and train them together using the same loss function to match the joint distribution $P(x_j, x_{j+1})$. Now, consider a bi-directed neighbor of X_{j+1}: $X_{j+2} \in bnb(X_{j+1})$. As they share another common confounder, $X_{j+1} \not\perp\!\!\!\perp X_{j+2}$. This implies that we need to feed a same confounding noise as input to both $f_{X_{j+1}}$ and $f_{X_{j+2}}$ and train them together to match the joint distribution $P(x_{j+1}, x_{j+2})$. However, $X_j \leftrightarrow X_{j+1}$ demands we train f_{X_j} and $f_{X_{j+1}}$ together and $X_{j+1} \leftrightarrow X_{j+2}$ demands that we train $f_{X_{j+1}}$ and $f_{X_{j+2}}$ together. Thus, to preserve both dependency we have to train all $f_{X_j}, f_{X_{j+1}}$ and $f_{X_{j+2}}$ together. This set gradually expands to the whole c-component S_i, i.e, all variables that are connected with bi-directed edges. Thus if we train mechanisms for all variables in the c-component together, and do the same for all c-components $\{S_1, S_2, ..., S_k\}$, we can match the joint $P(v)$.

(2) To prove the converse statement, consider a causal graph that consist of a cycle of c-component:

$$X_1 \leftrightarrow X_2 \leftrightarrow ... \leftrightarrow X_n \leftrightarrow X_1;$$

We proved in (1.) that if we have a partition $\{S_1\} = \{X_1, ..., X_n\}$ where S_1 is a c-component then algorithm A is consistent, i.e., it fits the joint distribution correctly.

Now, suppose, we can modularize the training process further based on any partition $\{S_1, S_2, ..., S_k\}$ of the c-component $\{X_1, ..., X_n\}$ where $k > 1$.

Note that each variable X_i in this c-component cycle contains at least two unobserved confounders as parents. Without loss of generality, lets assume that $S_i = \{X_i\}$. If we want to train S_i modularly; separated from rest, we have to break atleast one of the neighbor of X_i in the c-component cycle. This implies that, f_{X_i} receives both U_{X_{i-1},X_i} and $U_{X_i,X_{i+1}}$ as inputs in the true SCM, but we are training a model to learn a proxy of f_{X_i} while giving it signal from one neighbor. So, there is no guarantee f_{X_i} will utilize both of the confounders.

No matter which subset S_i, we start with for modular training, the neural networks in X_i might ignore one of the unobserved confounder between $X_{i-1} \leftrightarrow X_i$ and $X_i \leftrightarrow X_{i+1}$ and use the other confounder attempting to match its dependence with all other variables, i.e, $P(S_i|S_{\pi_{i-1}})$ [1]. As we are training S_i modularly cut off from one of its neighbor, it will try to get all signals from the confounder associated with remaining neighbor. Further modularization than the c-component level will prevent X_i to learn proper dependence with both of the neighbors X_{i-1} and X_{i+1}. Thus, for the considered cyclic c-component graph, whatever modularization is performed and training order is adopted, the joint distribution will not fit.

□

B.1 ADDITIONAL DISCUSSION

Suppose, mechanisms in S_i are not trained together. First let us consider the case when any neighboring pair $X_j, X_{j+1} \in S_i$ are trained independently, i.e., f_{X_j} is trained independently from $f_{X_{j+1}}$. When they are trained together, the generative mechanisms are as follows:

$$\begin{aligned} x_j &= f_j(pa(x_j), u, \mathcal{U}_{X_j} \setminus \{u\} \\ x_{j+1} &= f_{j+1}(pa(x_{j+1}), u, \mathcal{U}_{X_{j+1}} \setminus \{u\} \end{aligned} \tag{14}$$

Here, U is the shared confounder. $\mathcal{U}_{X_j} \setminus \{u\}$ are remaining confounder that affects X_j and $\mathcal{U}_{X_{j+1}} \setminus \{u\}$ are remaining confounders that affect X_{j+1}.

When f_{X_j} and $f_{X_{j+1}}$ are trained independently, the generative mechanisms becomes as follows:

$$\begin{aligned} x_j &= f_j(pa(x_j), u', \mathcal{U}_{X_j} \setminus \{u'\} \\ x_{j+1} &= f_{j+1}(pa(x_{j+1}), u, \mathcal{U}_{X_{j+1}} \setminus \{u\} \end{aligned} \tag{15}$$

Here, both u' and u are sampled from the same distribution $P(u)$. As u' is fed as input during training of f_{X_j} which is separate from $f_{X_{j+1}}$, u and u' varies independently. This breaks the dependency between X_j and X_{j+1} as they are not controlled by the same noise values. Thus $P_\theta(x_j, x_{j+1}) \neq P(x_j, x_{j+1})$ and eventually $P_\theta(v) \neq P(v)$.

(2.2) Now consider that mechanisms of $X_1, ..., X_j, X_{j+1}, ..., X_m$ are trained sequentially one by one in this order. X_1 is trained as mentioned before (Equation 14):

$$x_1 = f_1(pa(x_1), u, \mathcal{U}_{X_1} \setminus \{u\} \tag{16}$$

After training f_1, we freeze its model weights. As both f_1 and f_2 share the common confounder U, we sample the same $u \sim P(u)$ and feed it to both f_1 and f_2. As f_1 is frozen, it will use the confounding noise for inference while f_2 will use it for training. We can match fake X_1, X_2 with real X_1, X_2 samples to update model weights of f_2.

$$x_2 = f_1(pa(x_2), u, \mathcal{U}_{X_2} \setminus \{u\} \tag{17}$$

[1] here, π_{i-1} is all variables in the topological order before S_i

Suppose both $\{X_1, X_3\} \subseteq bnb(X_2)$ and the shared confounder between X_1 and X_2 is U while the confounder between X_2, X_3 is U'. Now, as we have already trained f_1 but not f_3 yet, while training f_2, even if we feed both U and U' as input to f_2, there is no guarantee that f_2 would use U' at all. There might be a possibility that f_2 might utilize only U to match the joint distribution $P_\theta(x_1, x_2) = P(x_1, x_2)$ but not $P_\theta(x_1, x_2, x_3) = P(x_1, x_2, x_3)$. If we had f_3 as trained, we could freeze both f_1 and f_3, feed them U and U' as well for inference and match $P_\theta(x_1, x_2, x_3)$ with $P(x_1, x_2, x_3)$ to update model weights of f_2. However, to train f_3, we would need f_2 as pre-trained which creates a cyclic situation. Thus, eventually, sequentially training fails to match the joint distribution $P(v)$.

C ALGORITHMS

Algorithm 1 Fed-DCM Algorithm

Input: Dataset $\mathcal{D}$, Causal graph $\mathcal{G}$, Variables $\mathbf{V} = \{V_1, V_2, \ldots, V_n\}, n = |\mathbf{V}|$
Client initialization:
for each $V \in \mathbf{V}$ **do**
    Initialize weights of $f_V(Pa(V), U_V)$ as $w[V]$
$[\mathbf{S}_i, \text{Pa}(\mathbf{S}_i)] \leftarrow$ c_component_partition($\mathcal{G}$)
$C_g(G) =$ Find c-component $S \in \{S\}_i$ s.t. $\mathcal{S}_{user} \in S$
Server executes:
Initialize model weights $w_0[V]$; for all $V \in S_{\mathcal{V}'}$.
for each round $t = 1, 2, \ldots$ **do**
    $C_t \leftarrow$ (random set of $max(\alpha C, 1)$ clients)
    for each client $k \in C_t$ **in parallel do**
        $w_{t+1}^k \leftarrow$ CLIENTUPDATE(k, w_t)
    for each variable $V \in S_{\mathcal{V}'}$ **do**
        $w_{t+1}[V] \leftarrow \sum_{k=1}^{K} \frac{n_k}{n} w_{t+1}^k[V]$

LocalTraining$(w, \mathcal{B}, S, Pa(S))$: *(Models in a cc)*
for each local epoch i from 1 to E **do**
    for each batch $b \in \mathcal{B}$ **do**
        Sample $pa(\mathbf{S}_i) \sim$ Uniform(support(Pa($\mathbf{S}_i$)))
        $D^R[\mathbf{S}] =$ getRealIntvData$(b, \mathcal{G}, \mathbf{S_i}, pa(\mathbf{S_i}))$
        $D^F[\mathbf{S}] =$ getFakeIntvData$(f_{V_i \in \mathbf{V}}, \mathbf{S_i}, pa(\mathbf{S_i}))$
        $\ell =$ dist(D^F, D^R)
        for each $V \in S$ **do**
            $w[V] \leftarrow w[V] - \eta \nabla \ell$
return w

ClientUpdate(c, w)**:** *(Run on client c)*
$\mathcal{B} \leftarrow$ (split D^c into batches of size B)
for each $S \in \{S_i\}_i \setminus \{S_{\mathcal{V}'}\}$ **do**
    $w =$ **LocalTraining**$(w, \mathcal{B}, S_i, Pa(S_i))$ **in parallel**
    Save $\{w[V]\}_{V \in S}$ locally.
$w =$ **LocalTraining**$(w, \mathcal{B}, S_{\mathcal{V}'}, Pa(S_{\mathcal{V}'}))$ **in parallel**
return $\{w[V]\}_{V \in S_{\mathcal{V}'}}$ to server

Algorithm 2 getRealIntvData$(\mathcal{D}, \mathcal{G}, \mathbf{S}, pa(\mathbf{S}))$

Input: Dataset $\mathcal{D}$, Causal graph $\mathcal{G}$, C-component $\mathbf{S}$, blanket $Pa(\mathbf{S})$.
$An = V_{\pi^{j-1}} \cap (S_i \cup pa(S_i))$; $\pi_{\mathcal{G}}$ be the ancestral order.
for each $V_j \in \mathbf{S}$ **do**
    Train $M_j(An)$ on $\mathcal{D}$ such that $M_j(An) \sim P(v_j|An)$
Fix $pa(\mathbf{S})$ in $M_{j:V_j \in \mathbf{S}}$ and ancestral sample to obtain $D^R[\mathbf{S}] \sim P(\mathbf{S} \mid \text{do}(\text{Pa}(\mathbf{S})))$
Return $D^R[\mathbf{S}]$

Algorithm 3 getFakeIntvData($\mathbb{G}_{V_i \in \mathbf{V}}, \mathbf{S}, pa(\mathbf{S})$)

1: **Input:** DCM $\mathbb{G}_{V_i \in \mathbf{V}}$, C-component $\mathbf{S}$, blanket $Pa(\mathbf{S})$.
2: Fix $pa(\mathbf{S})$ in $\mathbb{G}_{V_i \in \mathbf{V}}$ and ancestral sample to obtain $D^F[\mathbf{S}] \sim P(\mathbf{S} \mid \text{do}(\text{Pa}(\mathbf{S})))$
3: **Return** $D^F[\mathbf{S}]$

MATHEMATICAL NOTATION

The table below lists and defines the mathematical symbols used throughout this paper:

Symbol	Description
$\mathcal{I}$	The given/detected variable heterogeneous across clients.
$\mathcal{C}$	Set of all clients
$C(G)$	Set of all c-components
$C_l(G)$	Set of c-components trained using only local data
$C_g(G)$	Set of c-components trained using
S_i	i-th c-component in the graph after factorization
$\mathbb{G}$	DCM generators
$\mathcal{S}_{user}$	Client proposed set
$\mathcal{S}_{FL}$	Set of mechanism that we select for federated learning.
$\mathbf{M}$	Global models
$\mathcal{M}$	SCM
f^*	True mechanism of SCM
f	Mechanism of DCM
$\mathcal{V}$	Set of all causal variables in a structural causal model
$\nabla f(x)$	Gradient of $f(x)$
∂	Partial derivative