Learning Probabilities of Causation with Mask-Augmented Data

Shuai Wang [1] **Yizhou Sun** [2] **Judea Pearl** [2] **Ang Li** [1]

Abstract

Probabilities of causation play a central role in modern decision making. Tian and Pearl first introduced formal definitions and derived tight bounds for three binary probabilities of causation, such as the probability of necessity and sufficiency (PNS). However, estimating these probabilities requires both experimental and observational distributions specific to each subpopulation, which are often unreliable or impractical to obtain from limited population-level data. To solve this problem, we propose two machine learning models: Exact-MLP and Mask-MLP, which are trained on a small set of reliable subpopulations and are able to predict PNS bounds for all other subpopulations. We validate our models across four Structural Causal Models (SCMs), each evaluated on population-level data with sample sizes between 100k and 200k. Our models achieve average mean absolute errors (MAEs) of roughly 0.03 on main tasks, reducing MAE by about 80% relative to the corresponding baselines. These results demonstrate both the feasibility of machine learning models for learning probabilities of causation and the effectiveness of the proposed approach.

1. Introduction

Understanding causal relationships and estimating probabilities of causation are crucial in fields such as healthcare, policy evaluation, and economics (Pearl, 2009; Imbens & Rubin, 2015; Heckman & Pinto, 2015). Unlike correlation-based methods, causal inference enables decision-makers to determine whether an action or intervention directly leads to a desired outcome. This is particularly essential in personalized medicine, where accurately assessing treatment effects ensures both efficacy and safety (Mueller & Pearl, 2023). Moreover, causal reasoning enhances machine learning applications by improving accuracy (Li et al., 2020), interpretability, and fairness (Plecko & Bareinboim, 2022) in automated decision making. Despite its broad significance, estimating probabilities of causation remains challenging due to data **limitations**. In this paper, we address this challenge by leveraging machine learning techniques to predict probabilities of causation for subpopulations with **insufficient** data.

The study of probabilities of causation began around 2000, when researchers introduced three probabilities of causation including PNS within the framework of SCMs (Pearl, 1999; Galles & Pearl, 1998; Halpern, 2000; Pearl, 2009). In this paper, we focus on PNS, since it is most complex and representative. Specifically, PNS represents the chance that the cause is both required and sufficient for the outcome. And PNS is formally defined in Section 3; its practical value lies in helping us make informed, causality based decisions. For example, while taking Tylenol (a fever reducing drug) is correlated with having a fever, we don't give Tylenol because of the correlation. Instead, we give them because of a causal belief (PNS): without the drug, fever would likely persist; with the drug, it would likely subside.. Importantly, traditional randomized controlled trials (RCTs), though powerful for estimating average treatment effects, are not always sufficient for answering such counterfactual questions as illustrated in (Li & Pearl, 2019).

Subsequently, (Tian & Pearl, 2000) derived tight bounds for these probabilities using Balke's linear programming (Balke, 1995), incorporating both observational and experimental data[1]. Nearly two decades later, (Li & Pearl, 2019) formally proved these bounds and introduced the unit selection model, a decision making framework based on the linear combination of probabilities of causation. More recently, (Li & Pearl, 2024b) extended the definitions and bounds to a more general form. Additionally, (Mueller et al., 2022), as well as (Dawid et al., 2017), demonstrated that these bounds could be further refined given specific causal structures.

[1]Florida State University, Tallahassee, FL 32306, USA [2]University of California Los Angeles, Los Angeles, CA 90095, USA. Correspondence to: Shuai Wang <sw23v@cs.fsu.edu>, Yizhou Sun <yzsun@cs.ucla.edu>, Judea Pearl <judea@cs.ucla.edu>, Ang Li <angli@cs.fsu.edu>.

Preprint. March 7, 2026.

[1]Observational data refers to data typically collected through surveys, where individuals can access and freely choose whether to receive a treatment. In contrast, experimental data is usually obtained from randomized controlled trials, where individuals are randomly assigned to receive or not receive the treatment.

However, all of the above estimators of probabilities of causation require sufficiently large observational and experimental samples. Moreover, following (Li et al., 2022), obtaining reliable estimates for a subpopulation typically needs on the order of 300 experimental and 300 observational samples. While computing PNS bounds for a small number of well-sampled subpopulations is feasible, producing reliable bounds for *all* subpopulations quickly becomes statistically and computationally intractable.

Exact subpopulations correspond to fully specified covariate profiles (all covariates are 0 or 1), resulting in 2^{10}=1024 subpopulations. Mask-augmented subpopulations allow some covariates to be unspecified (denoted by X), yielding 3^{10}=59,049 subpopulations. We refer to queries over exact subpopulations as **exact queries**, and queries over mask-augmented subpopulations as **mask queries**. Note that exact queries are a special case of mask queries.

In this setting, direct plug-in computation of PNS becomes impractical: approximately 17% of mask-augmented subpopulations and 99% of exact subpopulations have no *valid* data after enforcing the per-group sample requirement in *both* datasets, and among the remaining exact subpopulations, the MAE ranges from 0.1499 to 0.2076, which is inadequate for practical use.

Hence, in this research, we will demonstrate the potential of machine learning models to achieve accurate estimations for subpopulations with insufficient data. Our approach first computes PNS bounds for both exact and mask-augmented subpopulations with sufficient data using causal formulas. These bounds are then used to train two Multilayer perceptron (MLP) based predictors (Rumelhart et al., 1986): the model trained on the mask-augmented query space is called **Mask-MLP**, while the model trained on exact queries is a standard MLP and is referred to as **Exact-MLP** for distinction. We evaluate model performance on test data using oracle PNS bounds derived from causal equations under a known data-generating process. To assess prediction quality comprehensively, we evaluate our models across four distinct SCMs and report results averaged over five runs.

1.1. Contributions

- **Learning to predict PNS bounds for subpopulations without enough data:** We formalize PNS bounds prediction as a supervised learning problem where only a small subset of subpopulations have sufficient experimental and observational data. Therefore, our models can predict the PNS bounds for all subpopulations.
- **Mask-augmented subpopulation modeling that scales supervision from 2^d to 3^d:** We introduce a mask-augmented representation of subpopulation queries and train a single *Mask-MLP* on 3^d mask-augmented subpopulations. This model not only predicts mask queries, but also achieves better performance on the same exact query prediction task under data scarcity.

2. Related Work

As the demand for explainable AI and causal reasoning grows, recent studies increasingly combine causal inference and machine learning. Broadly, the literature falls into two lines: (i) leveraging causal principles to improve ML models; and (ii) using ML to estimate or extrapolate causal quantities.

Causal principles for ML. By incorporating counterfactual reasoning, (Kusner et al., 2017) proposed a framework for counterfactual fairness. A number of explanation methods operationalize the notions of necessity and sufficiency within a predictive model. For example, (Galhotra et al., 2021) (LEWIS) and (Watson et al., 2021) (LENS) introduce scores inspired by "necessary/sufficient causes" to highlight influential features. However, their operationalizations (e.g., NeSuf) are defined with respect to the *model's* behavior rather than the structural counterfactual semantics of Pearl's PNS; hence they are not equivalent to PNS under standard SCM definitions. These approaches aim primarily at improving explainability of ML predictions rather than estimating causal quantities per se.

ML for causal quantities. (Xia et al., 2022) (GAN-NCM) learns neural SCMs to approximate counterfactuals from observational and experimental data, but typically assumes access to all relevant features and sufficient data for reliable SCM fitting. In contrast, our setting targets *subpopulation-level* probabilities of causation (e.g., PNS bounds) when many subpopulations are sparsely observed or even unobserved. Classical identification and bounding results (e.g., Tian–Pearl bounds and related LP formulations) provide analytic or optimization-based constraints. Our contribution is complementary: given reliable bounds on a small set of subpopulations, we learn to extrapolate *feasible and calibrated* PNS bounds to long-tail subpopulations under data scarcity, and evaluate them by feasibility, coverage, and policy-oriented ranking metrics.

3. Preliminaries

In this section, we review the fundamental concepts of causal inference necessary for understanding the rest of the paper. We begin by discussing the definition of PNS as introduced by (Pearl, 1999), followed by the tight bounds of PNS (Tian & Pearl, 2000). Experienced readers may skip this section.

Similar to the works mentioned above, we adopt the causal

language of Structural Causal Models (SCMs) (Galles & Pearl, 1998; Halpern, 2000). In this framework, the counterfactual statement "Variable Y would have the value y had X been x" is denoted as $Y_x = y$, abbreviated as y_x. We consider two types of data: experimental data, expressed as causal effects $P(y_x)$, and observational data, represented by the joint probability function $P(x, y)$. Unless otherwise specified, we assume X and Y are binary variables in a causal model M, with x and y denoting the propositions X = true and Y = true, respectively, and x' and y' representing their complements. For simplicity, we focus on binary treatments and effects; extensions to multi-valued cases are discussed by (Pearl, 2009) (p. 286, footnote 5) and (Li & Pearl, 2024a).

First, the definition of PNS, is defined using SCM as follow (Pearl, 1999):

Definition 3.1 (Probability of necessity and sufficiency (PNS))**.** Let X and Y be two binary variables in a causal model M, let x and y stand for the propositions $X = true$ and $Y = true$, respectively, and x' and y' for their complements. The probability of necessity and sufficiency is defined as the expression

$$\begin{aligned} \text{PNS} &\triangleq P(Y_{X=true} = true, Y_{X=false} = false) \\ &\triangleq P(y_x, y'_{x'}). \end{aligned}$$

Then, the oracle bounds of PNS is given by (Tian & Pearl, 2000).

$$\max \left\{ \begin{array}{c} 0, \\ P(y_x) - P(y_{x'}), \\ P(y) - P(y_{x'}), \\ P(y_x) - P(y) \end{array} \right\} \leq \text{PNS}, \tag{1}$$

$$\min \left\{ \begin{array}{c} P(y_x), \\ P(y'_{x'}), \\ P(x, y) + P(x', y'), \\ P(y_x) - P(y_{x'}) + \\ P(x, y') + P(x', y) \end{array} \right\} \geq \text{PNS}. \tag{2}$$

The advantage of PNS, as illustrated by (Li & Pearl, 2019), lies in capturing the portion of effects attributable specifically to the treatment, rather than the total causal effect. The primary objective of this paper is then to predict Equations 1 and 2 (i.e., the lower and upper bounds of the PNS) for any subpopulations using those with sufficient data (i.e., sufficient data to estimate the distributions $P(X, Y)$ and $P(Y_X)$.) Due to space constraints, the focus will be on the bounds of PNS (i.e., Equation 1 and 2). Unless otherwise specified, the discussion will be limited to binary treatment and effect, meaning both X and Y are binary.

4. Main Framework

4.1. The Machine Learning Pipeline

Our framework consists of two stages, summarized in Figure 1. Firstly, from an SCM we compute oracle PNS bounds via Eqs. (1)–(2) as test label, and derive sample-based bounds from finite experimental/observational data (after threshold filtering) as training labels. We then train predictors under either exact supervision on $q \in \{0, 1\}^d$ or mask-augmented supervision on $q \in \{0, 1, X\}^d$, and evaluate by MAE against the oracle bounds on exact and masked queries.

4.2. Causal Dataset Generation

SCMs and subpopulation queries. We generate synthetic data from four SCMs shown in Figure 2 (full structural equations and additional details are provided in the appendix). Our goal is to predict PNS bounds for both exact and mask queries defined over d observed binary covariates. Exact queries $q \in \{0, 1\}^d$ fully specify a covariate profile, whereas mask queries take the form $q \in \{0, 1, X\}^d$, where X denotes an unspecified covariate value. For example, $q = (1, X, 0)$ represents the union of the two exact profiles obtained by setting the second entry to 0 or 1. Exact queries are a special case of mask-augmented queries.

Oracle test set. To evaluate model with accurate bounds, we compute *oracle* causal quantities from SCM by causal equation. For any query q, we compute oracle experimental probability $P(y_x \mid q)$ and $P(y_{x'} \mid q)$ as well as the oracle observational joint distribution $P(x, y \mid q)$ by marginalizing over all hidden variables in the SCM (details in the appendix). These oracle distributions are then plugged into the Tian–Pearl bounds in Eqs. (1)–(2) to obtain the oracle lower and upper bounds of PNS for query q, which serve as the **test labels**. And a mask query $q \in \{0, 1, X\}^d$ corresponds to a collection of fully specified covariate vectors $z \in \{0, 1\}^d$ obtained by filling in the masked entries (those with X). Let $\mathcal{Z}(q)$ denote the set of such exact covariate vectors consistent with q, and define the normalizer

$$\pi(q) := \sum_{z \in \mathcal{Z}(q)} P(z). \tag{3}$$

Oracle quantities conditioned on q are computed by mixture aggregation over $\mathcal{Z}(q)$:

$$\begin{aligned} P(y_x \mid q) &= \frac{1}{\pi(q)} \sum_{z \in \mathcal{Z}(q)} P(y_x \mid z)\, P(z), \\ P(x, y \mid q) &= \frac{1}{\pi(q)} \sum_{z \in \mathcal{Z}(q)} P(x, y \mid z)\, P(z). \end{aligned} \tag{4}$$

The same aggregation applies to $P(y_{x'} \mid q)$. Finally, we

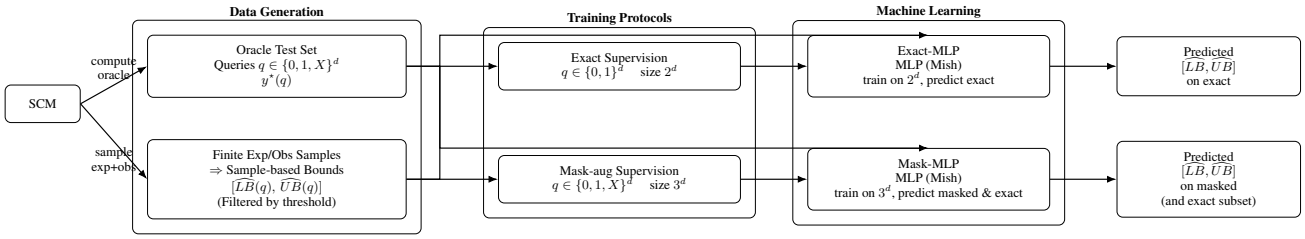
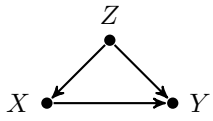
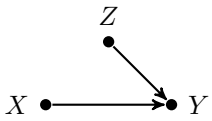


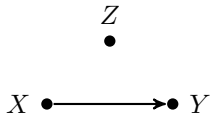
Figure 1. Pipeline for predicting PNS bounds under exact and mask queries. Oracle test labels are computed from the SCM, while training labels are sample-based bounds estimated from finite experimental and observational data.



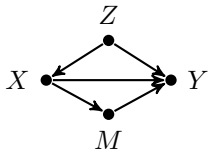
(a) Confounder Z with direct effect



(b) Covariate Z



(c) Unrelated Z with Direct effect



(d) Mediator M and direct effect

Figure 2. Different SCMs in this study.

obtain oracle PNS bounds for q by plugging the aggregated oracle quantities into Eqs. (1)–(2).

Generate samples for training labels. To emulate realistic data scarcity, we need to generate a finite number of observational and experimental samples from the SCM. For the observational setting, (X, Y) are generated according to the causal equations. And for the experimental setting, X is randomized and the remaining variables are generated by the causal equations. Only observed covariates and outcomes are retained; all hidden variables are unobserved. Note that all uncertainty are caused by those unobserved variables, which means if all variables are observed, the PNS would be an accurate value rather than bounds.

Calculate bounds for training samples Training labels are computed from finite samples by estimating. For each query q, we obtain the experimental distribution $P(y_x \mid q)$ and $P(y_{x'} \mid q)$ from the experimental samples and the observational distribution $P(x, y \mid q)$ from the observational samples. Then we plugging these empirical estimates into Eqs. (1)–(2). To ensure feasibility, we add a query q into the training set only if it has sufficient support in *both* experimental and observational samples (e.g., at least 300 samples each (Li et al., 2022)) with constraints (e.g., $\mathrm{LB} \leq \mathrm{UB}$). This yields two supervised datasets: (i) an **exact** training set over $q \in \{0,1\}^d$ (used for Exact-MLP), and (ii) a **mask-augmented** training set over $q \in \{0,1,X\}^d$ (used for Mask-MLP).

4.3. Machine Learning

We study whether a multilayer perceptron (MLP) can predict PNS bounds to subpopulation queries with insufficient data. Given the considerable number of zero values in the data, we proposed using $\mathrm{Mish}(s) = s \cdot \tanh(\ln(1 + e^s))$ (Misra, 2019), as a smooth and stable activation function rather than ReLU, to train two independent regressors for the lower and upper bounds.

Input representation. Each query $q \in \{0,1,X\}^d$ is encoded by a $3d$-dimensional one-hot vector. For each coordinate $j \in [d]$ and each symbol $c \in \{0,1,X\}$, define

$$\phi_{j,c}(q) = \mathbb{I}[q_j = c]. \tag{5}$$

We then form the feature vector $\phi(q) \in \{0,1\}^{3d}$ by concatenating these indicators over all (j,c) pairs in a fixed order:

$$\phi(q) = \mathrm{concat}\big(\{\phi_{j,c}(q)\}_{j\in[d],\, c\in\{0,1,X\}}\big). \tag{6}$$

Predictors. We train two independent MLP regressors, f^{LB} and f^{UB}, to predict the lower and upper bounds, respectively. Both networks share the same architecture and use Mish activations; the output layer uses a sigmoid so that predictions lie in $[0,1]$. The training data are the sample-based bounds computed from finite experimental and observational data (Section 4).

Training protocols. Exact-MLP (exact supervision). We train f^{LB} and f^{UB} on exact queries only, i.e., $q \in \{0,1\}^d$ (up to 2^d queries after thresholding), and evaluate on oracle bounds for exact queries.

Mask-MLP (mask-augmented supervision). We train the same architecture on the enlarged query space $q \in \{0,1,X\}^d$ (up to 3^d queries after thresholding). The resulting model is evaluated on oracle bounds for both mask queries and the exact-query subset.

Algorithm 1 Exact-MLP

Require: SCM M with d observed binary covariates; budgets $(N_{\text{exp}}, N_{\text{obs}})$; threshold grid $\mathcal{T}$.
Ensure: Predictors and MAE on oracle PNS bounds over $\mathcal{Q}_{\text{exact}} = \{0,1\}^d$.
Sample experimental data $\mathcal{D}_{\text{exp}}$ (size N_{exp}) and observational data $\mathcal{D}_{\text{obs}}$ (size N_{obs}) from M.
Compute oracle bounds $\{(LB^\star(q), UB^\star(q))\}_{q\in\mathcal{Q}_{\text{exact}}}$ via Eqs. (1)–(2).
for $\tau \in \mathcal{T}$ **do**
  Initialize training set $\mathcal{D}_\tau \leftarrow \emptyset$.
  for $q \in \mathcal{Q}_{\text{exact}}$ **do**
    Estimate experimental distribution on q from $\mathcal{D}_{\text{exp}}$:
      $\widehat{P}(y_x \mid q) = \widehat{P}(Y{=}1 \mid do(X{=}1), q)$
      $\widehat{P}(y_{x'} \mid q) = \widehat{P}(Y{=}1 \mid do(X{=}0), q)$
    Estimate the observational distribution table on q from $\mathcal{D}_{\text{obs}}$:
      $\widehat{P}(X, Y \mid q)$ (the full 2×2 table over $X, Y \in \{0,1\}$).
    Compute sample-based bounds $[\widehat{LB}_\tau(q), \widehat{UB}_\tau(q)]$ via Eqs. (1)–(2).
    Compute supports $\text{support}_{\text{exp}}(q)$ in $\mathcal{D}_{\text{exp}}$ and $\text{support}_{\text{obs}}(q)$ in $\mathcal{D}_{\text{obs}}$.
    if $\text{support}_{\text{exp}}(q) \geq \tau$ **and** $\text{support}_{\text{obs}}(q) \geq \tau$ **and** $0 \leq \widehat{LB}_\tau(q) \leq \widehat{UB}_\tau(q) \leq 1$ **then**
      Add $(\phi(q), \widehat{LB}_\tau(q), \widehat{UB}_\tau(q))$ to $\mathcal{D}_\tau$.
    end if
  end for
  Train two regressors f_τ^{LB} and f_τ^{UB} (MLP+Mish) on $\mathcal{D}_\tau$.
  Predict for all $q \in \mathcal{Q}_{\text{exact}}$:
    $\widehat{LB}(q) = f_\tau^{LB}(\phi(q))$, $\widehat{UB}(q) = f_\tau^{UB}(\phi(q))$.
  Record $\text{MAE}_{LB}(\tau) = \mathbb{E}_{q\in\mathcal{Q}_{\text{exact}}}\left|\widehat{LB}(q) - LB^\star(q)\right|$, $\text{MAE}_{UB}(\tau) = \mathbb{E}_{q\in\mathcal{Q}_{\text{exact}}}\left|\widehat{UB}(q) - UB^\star(q)\right|$.
end for
Report the best τ (or the full sweep over $\mathcal{T}$) and the corresponding predictors.

Evaluation. We compare the predicted bounds to oracle PNS bounds computed from the SCM and report the average MAEs for both the lower and upper bounds. Since the threshold setting affects the model MAEs, to evaluate different models fairly, we test the thresholds from 100 to 2000, and select the best MAE and its corresponding threshold to represent the performance. The reported MAEs are averaged across five runs.

4.4. Algorithm

Through the whole process, we can summarize the process into two algorithms as 1 and 2. In a word, for each threshold τ, the algorithm will calculate the oracle PNS bounds and sample-based bounds by formula 1 and 2. Then the PNS bounds would be added to training data $\mathcal{D}_\tau$ iff this subgroup contains enough legal samples (larger than τ). Next, we train the MLP f_τ^{LB} and f_τ^{UB}, and obtain the MAEs of the MLP compared with the oracle PNS bounds. Finally, we obtain the MAEs through different thresholds τ.

Algorithm 2 Mask-MLP

Require: SCM M with d observed binary covariates; budgets $(N_{\text{exp}}, N_{\text{obs}})$; threshold grid $\mathcal{T}$.
Ensure: Predictors and MAE on oracle PNS bounds over $\mathcal{Q}_{\text{mask}} = \{0,1,X\}^d$ and $\mathcal{Q}_{\text{exact}} \subset \mathcal{Q}_{\text{mask}}$.
Sample experimental data $\mathcal{D}_{\text{exp}}$ (size N_{exp}) and observational data $\mathcal{D}_{\text{obs}}$ (size N_{obs}) from M.
Compute oracle bounds $\{(LB^\star(q), UB^\star(q))\}_{q\in\mathcal{Q}_{\text{mask}}}$ via Eqs. (1)–(2).
For masked queries, aggregate oracle quantities as in Eq. (4).
for $\tau \in \mathcal{T}$ **do**
  Initialize training set $\mathcal{D}_\tau \leftarrow \emptyset$.
  for $q \in \mathcal{Q}_{\text{mask}}$ **do**
    Estimate experimental distribution on q from $\mathcal{D}_{\text{exp}}$:
      $\widehat{P}(y_x \mid q) = \widehat{P}(Y{=}1 \mid do(X{=}1), q)$
      $\widehat{P}(y_{x'} \mid q) = \widehat{P}(Y{=}1 \mid do(X{=}0), q)$
    Estimate the observational distribution table on q from $\mathcal{D}_{\text{obs}}$:
      $\widehat{P}(X, Y \mid q)$ (the full 2×2 table).
    For masked q, obtain these estimates by aggregating exact-profile estimates as in Eq. (4).
    Compute sample-based bounds $[\widehat{LB}_\tau(q), \widehat{UB}_\tau(q)]$ via Eqs. (1)–(2).
    Compute supports $\text{support}_{\text{exp}}(q)$ and $\text{support}_{\text{obs}}(q)$.
    if $\text{support}_{\text{exp}}(q) \geq \tau$ **and** $\text{support}_{\text{obs}}(q) \geq \tau$ **and** $0 \leq \widehat{LB}_\tau(q) \leq \widehat{UB}_\tau(q) \leq 1$ **then**
      Add $(\phi(q), \widehat{LB}_\tau(q), \widehat{UB}_\tau(q))$ to $\mathcal{D}_\tau$.
    end if
  end for
  Train two regressors g_τ^{LB} and g_τ^{UB} (MLP+Mish) on $\mathcal{D}_\tau$.
  Predict for all $q \in \mathcal{Q}_{\text{mask}}$:
    $\widehat{LB}(q) = g_\tau^{LB}(\phi(q))$, $\widehat{UB}(q) = g_\tau^{UB}(\phi(q))$.
  Record MAEs on both spaces:
    $\text{MAE}_{LB}^{\text{mask}}(\tau) = \mathbb{E}_{q\in\mathcal{Q}_{\text{mask}}}\left|\widehat{LB}(q) - LB^\star(q)\right|$
    $\text{MAE}_{UB}^{\text{mask}}(\tau) = \mathbb{E}_{q\in\mathcal{Q}_{\text{mask}}}\left|\widehat{UB}(q) - UB^\star(q)\right|$
    $\text{MAE}_{LB}^{\text{exact}}(\tau) = \mathbb{E}_{q\in\mathcal{Q}_{\text{exact}}}\left|\widehat{LB}(q) - LB^\star(q)\right|$
    $\text{MAE}_{UB}^{\text{exact}}(\tau) = \mathbb{E}_{q\in\mathcal{Q}_{\text{exact}}}\left|\widehat{UB}(q) - UB^\star(q)\right|$
end for
Report the best τ (or the full sweep over $\mathcal{T}$) and the corresponding predictors.

5. Experimental Results

5.1. Experimental Setup

We evaluate on four SCMs (Confounder, Covariate, Direct, and Mediator) as in Figure 2. The number of observed covariates d is set to 10. Thereby, we have such two query spaces: (i) exact queries $q \in \{0,1\}^d$ (size $2^{10} = 1024$), and (ii) mask queries $q \in \{0,1,X\}^d$ (size $3^{10} = 59{,}049$), where X denotes an unspecified covariate. Oracle PNS bounds are computed by using the Tian–Pearl formulas in Eqs. (1)–(2). For mask queries, oracle quantities are aggregated as in Eq. (4) before applying Eqs. (1)–(2). To evaluate the relation between data budget and model performance, we generate finite experimental and observational samples under budgets

from 50k to 500k and construct sample-based training labels after threshold filtering (Section 4). We run five independent trials (random seeds 1–5) and report MAE for both the lower and upper bounds. Our main predictors are Exact-MLP (trained on exact subpopulations) and Mask-MLP (trained on mask-augmented subpopulations), both using the same MLP+Mish backbone (Section 4). While baseline computes sample-based bounds directly by Eqs. (1)–(2) for every possible subpopulation, and then remove bounds that are not feasible.

5.2. Overall Accuracy on Exact and Mask Queries

Table 1 summarizes MAE under both query spaces: the left panel reports performance on exact queries $q \in \{0,1\}^d$, and the right panel reports performance on the mask-augmented space $q \in \{0,1,X\}^d$. Across all SCMs and budgets, both MLP models substantially outperform the plug-in baseline. Overall, Mask-MLP is typically strongest on the exact subset as well, which indicates that training on the enlarged 3^d query space provides additional information and regularization. However, if we compare Exact-MLP with Mask-MLP on exact queries, there are two exceptions. The first occurs in the simplest Direct structure: Because the PNS bounds are fixed in the Direct structure, fewer training samples suffice, and Exact-MLP performs better for budgets above 100k. Second, their performances are close when the data budget is 500k, because this size can provide nearly enough training data for Exact-MLP. On the mask-augmented space, only Mask-MLP applies directly and it consistently improves over the baseline across SCMs and budgets, demonstrating that mask augmentation substantially expands the supported query space while retaining strong accuracy. In summary, both MLP models substantially outperform the baseline. Moreover, Mask-MLP is more robust under data scarcity and can predict PNS bounds for all mask-augmented subpopulations.

5.3. Threshold Sensitivity and Stability

We further analyze sensitivity to the threshold that filters low-support queries. Figure 3 shows representative threshold sweeps under the 200k budget (plots for other budgets are in Appendix A.4). There is a trade-off across threshold settings: a lower threshold yields more training data but also cause more noise due to small sample size, while a higher threshold implies better data quality but generate less training data. Accordingly, across SCMs, Mask-MLP usually performs better at higher thresholds, whereas Exact-MLP often prefers lower thresholds. And we select best choice of threshold to represent the performance of the model. In all cases, both models outperform the baseline at their respective best thresholds.

5.4. Case Study: Treat-or-Not for a Toxic but Potentially Life-Saving Drug

Background. We consider a life-threatening disease for which a drug exists but has substantial toxicity. Each patient is described by ten observed binary covariates $Z_{1:10}$ (e.g., two bits for age Z_1Z_2, sex Z_3, and comorbidities Z_4–Z_{10}), while additional unobserved factors may also influence outcomes. We observe 200k experimental (RCT) records and 200k observational records at the population level. Now consider a new patient whose covariate profile $z^\star$ is fully specified. The patient wishes to know whether to take the drug. However, like 936 of the 1024 exact subpopulations, the hospital has fewer than 300 recorded patients in this subpopulation, which is too limited for reliably estimating PNS bounds using the plug-in baseline. Let X denote treatment (x = treated, x' = untreated) and Y denote the clinical outcome (y = survived, y' = deceased). Beyond the observed $Z_{1:10}$, a mediator M (e.g., viral load) and further unobserved factors may also affect Y.

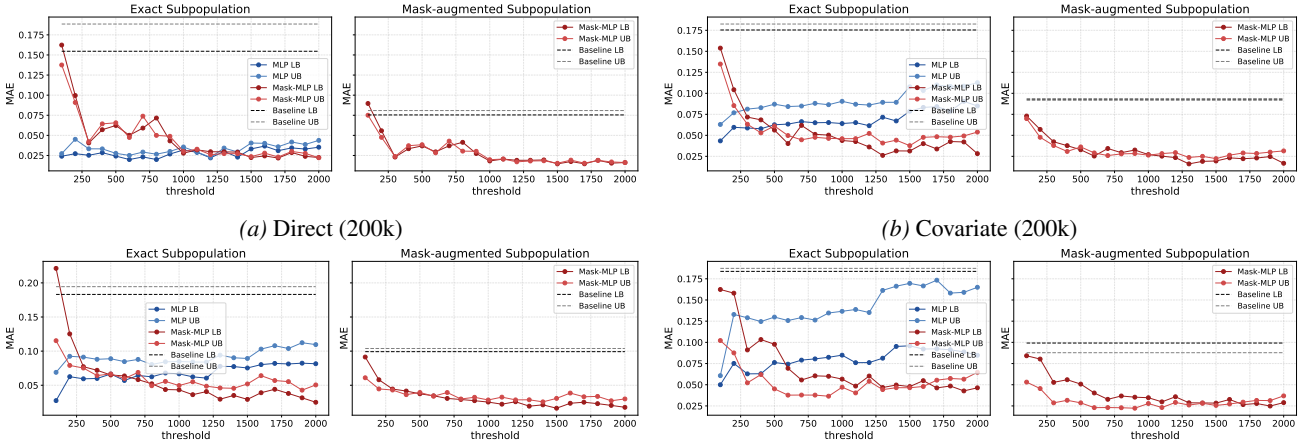
Pipeline. In this case, we have the number of observable covariates $d = 10$. Therefore, we got exact queries $q \in \{0,1\}^d$ and mask-augmented subpopulations $q \in \{0,1,X\}^d$. Next, following Section 4, we identify well-sampled subpopulations among the 2^{10} exact groups and compute their reliable sample-based bounds using Eqs. (1)–(2). We then train two predictors: (i) an **MLP (Mish)** trained on threshold-filtered exact supervision, and (ii) a **Mask-MLP (Mish)** trained on threshold-filtered mask-augmented supervision over the full 3^{10} query space. Both predictors extrapolate *feasible* PNS bounds to the long tail (including rare and unseen queries), outputting $\mathrm{PNS}_{\mathrm{LB}}(z^\star)$ and $\mathrm{PNS}_{\mathrm{UB}}(z^\star)$. At the same time, we also calculate PNS bounds via Eqs. (1)–(2) based on his limited subpopulation.

Decision rule. We set a decision threshold $\theta = 0.5$ (Treat iff $\mathrm{PNS} \geq 0.5$): we *treat* when $\mathrm{LB} \geq \theta$, *do not treat* when $\mathrm{UB} < \theta$, and mark *uncertain* otherwise. We abbreviate decisions as **T** (treat), **N** (not treat), and **U** (unknown).

Illustrative example under $\theta = 0.5$. Consider the patient with $z = 1101001110$ in Table 2. According to the oracle (true) bounds ($\mathrm{LB} = 0.2229$ and $\mathrm{UB} = 0.2746$), we have $\mathrm{UB} < 0.5$, which means the patient should **not be treated**. In contrast, the baseline plug-in method outputs $\mathrm{LB} = \mathrm{UB} = 1.0000$ under extreme data sparsity, and suggests **treatment** with full confidence. As for the MLP, it gives an uncertain conclusion, since its interval crosses the decision threshold ($\mathrm{LB} = 0.3529 < 0.5 < \mathrm{UB} = 0.5908$). Finally, Mask-MLP predicts $\mathrm{LB} = 0.1994$ and $\mathrm{UB} = 0.2890$, and correctly recommend **no treatment**, consistent with the oracle.

Table 1. MAE on oracle PNS bounds for exact queries $q \in \{0,1\}^d$ and mask queries $q \in \{0,1,X\}^d$ (Mask-MLP only). Boldface marks the lowest MAE in each panel (LB/UB).

SCM	Budget	Exact queries $q \in \{0,1\}^d$						Mask queries $q \in \{0,1,X\}^d$			
		Baseline		Exact-MLP (Mish)		Mask-MLP (Mish)		Baseline		Mask-MLP (Mish)	
		LB	UB	LB	UB	LB	UB	LB	UB	LB	UB
Direct	50k	0.1694	0.2089	0.0207	0.0255	**0.0188**	**0.0247**	0.0959	0.1168	**0.0139**	**0.0179**
Direct	100k	0.1499	0.1786	**0.0180**	**0.0181**	0.0196	0.0234	0.0859	0.0987	**0.0145**	**0.0159**
Direct	200k	0.1546	0.1885	**0.0199**	0.0227	0.0218	**0.0222**	0.0754	0.0809	**0.0145**	**0.0153**
Direct	500k	0.1268	0.1540	**0.0112**	**0.0096**	0.0204	0.0197	0.0606	0.0710	**0.0129**	**0.0129**
Covariate	50k	0.1958	0.2388	0.0623	0.0812	**0.0375**	**0.0507**	0.1095	0.1405	**0.0240**	**0.0294**
Covariate	100k	0.2076	0.1867	0.0606	0.0772	**0.0308**	**0.0360**	0.1157	0.1036	**0.0202**	**0.0221**
Covariate	200k	0.1754	0.1825	0.0434	0.0630	**0.0261**	**0.0377**	0.0922	0.0937	**0.0160**	**0.0223**
Covariate	500k	0.1541	0.1567	**0.0232**	0.0379	0.0287	**0.0299**	0.0750	0.0733	**0.0172**	**0.0172**
Confounder	50k	0.2166	0.2423	0.0572	0.0874	**0.0463**	**0.0667**	0.1337	0.1392	**0.0262**	**0.0388**
Confounder	100k	0.2072	0.2101	0.0575	0.0819	**0.0380**	**0.0500**	0.1165	0.1225	**0.0238**	**0.0288**
Confounder	200k	0.1831	0.1945	0.0275	0.0689	**0.0251**	**0.0428**	0.0992	0.1039	**0.0164**	**0.0257**
Confounder	500k	0.1523	0.1786	**0.0244**	**0.0382**	0.0307	0.0399	0.0752	0.0800	**0.0182**	**0.0219**
Mediator	50k	0.2089	0.2256	0.0849	0.1323	**0.0651**	**0.0725**	0.1230	0.1302	**0.0410**	**0.0483**
Mediator	100k	0.1917	0.2027	0.0731	0.1286	**0.0441**	**0.0506**	0.1175	0.1115	**0.0282**	**0.0306**
Mediator	200k	0.1838	0.1873	0.0500	0.0607	**0.0427**	**0.0365**	0.0992	0.0878	**0.0249**	**0.0223**
Mediator	500k	0.1564	0.1472	**0.0273**	0.0423	0.0301	**0.0221**	0.0755	0.0699	**0.0184**	**0.0132**



(a) Direct (200k) *(b)* Covariate (200k)

(c) Confounder (200k) *(d)* Mediator (200k)

Figure 3. Threshold sweeps at the 200k budget across four SCMs. Each plot contains two panels: evaluation on exact queries (left) and on mask queries (right). All MAEs are mean values over five runs.

Table 2. Decision at $\theta = 0.5$ for the rare exact group $z =$ `1101001110`. B: baseline (naive plug-in); M: MLP; MM: Mask-MLP.

	LB/UB				
z (10-bit)	True	B	M	MM	Dec. (True/B/M/MM)
`1101001110`	0.223/0.275	1.000/1.000	0.353/0.591	0.199/0.289	**N/T/U/N**

Robustness under a single missing feature. In practice, a patient may have missing or unclear feature (may caused by privacy issue or just unsure). We therefore construct ten *single-missing* variants by replacing exactly one coordinate of $z =$ `1101001110` with X (unknown). Table 3 reports the oracle bounds, the baseline predictions, and the Mask-MLP predictions. Among the ten single-missing variants, nine cases have the same output with exact subpopulation prediction, *only one* case becomes **unknown**. However, it is not caused by wrong predication: the oracle also has the same conclusion. This is the desired pattern: robustness

Table 3. Single-missing variants of $z = 1101001110$ under $\theta = 0.5$. Each row replaces one coordinate by X (unknown). We report oracle (True), baseline (B), and Mask-MLP (MM) LB/UB, plus decisions (T/N/U).

	LB/UB			
Masked query	True	B	MM	Dec. (True/B/MM)
X101001110	0.164/0.199	1.000/1.000	0.161/0.225	**N/T/N**
1X01001110	0.278/0.435	0.333/0.333	0.268/0.474	**N/N/N**
11X1001110	0.140/0.165	0.667/0.667	0.110/0.193	**N/T/N**
110X001110	0.308/0.491	0.361/0.444	0.294/0.451	**N/N/N**
1101X01110	0.324/0.528	0.324/0.532	0.382/0.528	**U/U/U**
11010X1110	0.170/0.205	0.000/0.000	0.170/0.227	**N/N/N**
110100X110	0.225/0.278	1.000/1.000	0.196/0.292	**N/T/N**
1101001X10	0.250/0.343	1.000/1.000	0.218/0.337	**N/T/N**
11010011X0	0.260/0.384	0.400/0.400	0.226/0.418	**N/N/N**
110100111X	0.222/0.273	1.000/1.000	0.196/0.286	**N/T/N**

to minor missingness, together with uncertainty flags if the query is intrinsically ambiguous.

6. Discussion

We studied whether our machine learning models can predict *oracle* PNS bounds for subpopulations with insufficient data based on finite experimental and observational samples. Across four SCMs and multiple sample budgets and the study case, we can draw such conclusion.: (i) learning-based predictors dramatically improve over naive plug-in estimation under limited data, and (ii) Mask-MLP typically strengthens generalization and further supports query space.

Model comparison and underlying reasons. Our most important issue is data size, or, more specifically, the sample budget. In the ideal scenario, assume that unlimited data can be provided; we can calculate PNS bounds via formulas based on oracle experimental and conditional probabilities. So, the baseline method is the best with sufficient data. If we consider the cost of data, the machine learning models could be better than the baseline. And the situation is also similar for the difference between Exact-MLP and Mask-MLP on exact subpopulation predictions. When the sample budget is relatively small for the SCM, Mask-MLP can be trained with much more mask-augmented data, which performs better. But when the data size is relatively large for the SCM, the MLP can learn from enough training data and is able to predict more accurately. In our case, when the sample budget is lower than 500k, Mask-MLP is more accurate and stable (except for the Direct SCM, which is too simple).

Decision making. In addition, our case study shows how this model can be applied in practice. In the case study, the conclusion computed directly using the Tian–Pearl formulas is completely opposite to the ground truth; Exact-MLP is uncertain, whereas Mask-MLP remains consistent with the ground truth. Moreover, when a single covariate is missing at decision time, only one of the ten single-missing variants is intrinsically ambiguous under the oracle (and this is also accurately identified by our model). This pattern is ideal for high-stakes settings: it is robust to minor missingness and raises uncertainty flags only when the underlying query itself has inherent ambiguity.

From full coverage to targeted decisions. A fundamental goal in causal decision making is to identify subpopulations with sufficiently large $\mathrm{PNS}_{\mathrm{LB}}$ (or small $\mathrm{PNS}_{\mathrm{UB}}$) rather than to perfectly predict every subpopulation. Hence, accurately predicting all subpopulations is unnecessary for practical use. An interesting direction is to determine a minimal training set that reliably recovers the top-k subpopulations under budget or risk constraints.

Limitations and future directions. Though our experiment is enough to show the availability of predicting PNS bounds via machine learning, we still leave a lot of space for more experiments. The first is data size for more complex SCMs: a 200k sample budget and Mask-MLP with Mish are enough to render good results in our SCM. But when the SCM is more complex, or more covariates are included, the sample budget may need to be changed. The second is the choice of ML models: although MLP is the best in our past experiment under our settings, for more complex SCMs, richer covariates, and some more complex model (like Transformer) may perform better than MLP. So, exploring more complex SCMs is important in the future.

An additional limitation is that on real-world data, oracle test bounds are generally unavailable, making it difficult to directly validate bound tightness and calibration beyond indirect or downstream evaluations. A potential way is to train and test only on reliable subpopulations.

7. Conclusion

In this paper, we show that PNS bounds can be learned and extrapolated to subpopulations with insufficient support from finite experimental and observational data. We propose two predictors, Exact-MLP and Mask-MLP, and empirically demonstrate that both consistently outperform naive plug-in estimation based on the Tian–Pearl bounds across four SCMs, on both exact subpopulation and mask queries. In low-data regimes, Mask-MLP typically achieves better accuracy than Exact-MLP, reflecting the benefit of mask-augmented supervision. This is also reflected in our examples: Mask-MLP showed great potential in real-world decision making based on PNS.

In the future, we can explore more complex SCMs and investigate whether more complex models would perform better. Furthermore, overcoming the lack of oracle data in real-world datasets is also a promising direction.

Acknowledgements

This research was supported in parts by grants from the National Science Foundation [#IIS-2106908 and #CNS-2321786], and Toyota Research Institute of North America [#PO-000897].

References

Balke, A. A. *Probabilistic counterfactuals: semantics, computation, and applications*. University of California, Los Angeles, 1995.

Dawid, P., Musio, M., and Murtas, R. The probability of causation. *Law, Probability and Risk*, 16:163–179, 2017.

Galhotra, S., Pradhan, R., and Salimi, B. Explaining black-box algorithms using probabilistic contrastive counterfactuals. *CoRR*, abs/2103.11972, 2021. URL https://arxiv.org/abs/2103.11972.

Galles, D. and Pearl, J. An axiomatic characterization of causal counterfactuals. *Foundations of Science*, 3(1):151–182, 1998.

Halpern, J. Y. Axiomatizing causal reasoning. *Journal of Artificial Intelligence Research*, 12:317–337, 2000.

Heckman, J. and Pinto, R. Causal analysis after haavelmo. *Econometric Theory*, 31(1):115–151, 2015.

Imbens, G. W. and Rubin, D. B. *Causal inference in statistics, social, and biomedical sciences*. Cambridge university press, 2015.

Kusner, M. J., Loftus, J., Russell, C., and Silva, R. Counterfactual fairness. *Advances in neural information processing systems*, 30, 2017.

Langley, P. Crafting papers on machine learning. In Langley, P. (ed.), *Proceedings of the 17th International Conference on Machine Learning (ICML 2000)*, pp. 1207–1216, Stanford, CA, 2000. Morgan Kaufmann.

Li, A. and Pearl, J. Unit selection based on counterfactual logic. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence*, 2019.

Li, A. and Pearl, J. Probabilities of causation with nonbinary treatment and effect. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 20465–20472, 2024a.

Li, A. and Pearl, J. Unit selection with nonbinary treatment and effect. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38(18), pp. 20473–20480, 2024b.

Li, A., J. Chen, S., Qin, J., and Qin, Z. Training machine learning models with causal logic. In *Companion Proceedings of the Web Conference 2020*, pp. 557–561, 2020.

Li, A., Mao, R., and Pearl, J. Probabilities of causation: Adequate size of experimental and observational samples. In *NeurIPS 2022 Workshop on Neuro Causal and Symbolic AI (nCSI)*, 2022.

Misra, D. Mish: A self regularized non-monotonic activation function. *arXiv preprint arXiv:1908.08681*, 2019.

Mueller, S. and Pearl, J. Perspective on 'harm' in personalized medicine – an alternative perspective. Technical Report R-530, Department of Computer Science, University of California, Los Angeles, CA, 2023. Forthcoming, American Journal of Epidemiology.

Mueller, S., Li, A., and Pearl, J. Causes of effects: Learning individual responses from population data. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence (IJCAI-22)*, pp. 2712–2718, 2022.

Pearl, J. Probabilities of causation: Three counterfactual interpretations and their identification. *Synthese*, pp. 93–149, 1999.

Pearl, J. *Causality*. Cambridge university press, 2nd edition, 2009.

Plecko, D. and Bareinboim, E. Causal fairness analysis. *arXiv preprint arXiv:2207.11385*, 2022.

Rumelhart, D. E., Hinton, G. E., and Williams, R. J. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, 1986. doi: 10.1038/323533a0.

Tian, J. and Pearl, J. Probabilities of causation: Bounds and identification. *Annals of Mathematics and Artificial Intelligence*, 28(1-4):287–313, 2000.

Watson, D., Gultchin, L., Taly, A., and Floridi, L. Local explanations via necessity and sufficiency: Unifying theory and practice, 2021. URL https://arxiv.org/abs/2103.14651.

Xia, K., Pan, Y., and Bareinboim, E. Neural causal models for counterfactual identification and estimation, 2022. URL https://arxiv.org/abs/2210.00035.

A. Appendix / supplemental material

A.1. The Causal Model

There are four SCMs mentioned in our work, which are Confounder, Covariate, Direct and Mediator. And their SCM are shown in figure 4.

(a) Confounder Z with direct effect

(b) Covariate Z

(c) Direct

(d) Mediator M and direct effect

Figure 4. Different SCMs in this study.

A.1.1. CONFOUNDER

Confounder SCM is shown in figure 4a. The coefficients for X_Z, Y_Z, and C_Y were uniformly generated from the range $[-1, 1]$, while the parameters of the Bernoulli distribution were uniformly generated from $[0, 1]$. The detailed model is as follows:

$$
\begin{cases}
Z_i & = U_{Z_i} \text{ for } i \in \{1, ..., 20\}, \\
X & = f_X(X_Z, U_X) \\
& = \begin{cases} 1 & \text{if } X_Z + U_X > 0.5 \\ 0 & \text{otherwise,} \end{cases} \\
Y & = f_Y(X, Y_Z, U_Y) \\
& = \begin{cases} 1 & \text{if } 0 < C_Y \cdot X + Y_Z + U_Y < 1 \\ 1 & \text{if } 1 < C_Y \cdot X + Y_Z + U_Y < 2 \\ 0 & \text{otherwise.} \end{cases}
\end{cases}
$$

where, U_{Z_i}, U_X, U_Y are binary exogenous variables with Bernoulli distributions.

$s.t.$,

$$
X_Z = \begin{bmatrix} Z_1 \; Z_2 \; ... \; Z_{20} \end{bmatrix} \times \begin{bmatrix} 0.259223510143 \\ -0.658140989167 \\ -0.75025831768 \\ 0.162906462426 \\ 0.652023463285 \\ -0.0892939586541 \\ 0.421469107769 \\ -0.443129684766 \\ 0.802624388789 \\ -0.225740978499 \\ 0.716621631717 \\ 0.0650682260309 \\ -0.220690334026 \\ 0.156355773665 \\ -0.50693672491 \\ -0.707060278115 \\ 0.418812816935 \\ -0.0822118703986 \\ 0.769299853833 \\ -0.511585391002 \end{bmatrix}, Y_Z = \begin{bmatrix} Z_1 \; Z_2 \; ... \; Z_{20} \end{bmatrix} \times \begin{bmatrix} -0.792867111918 \\ 0.759967136147 \\ 0.55437722369 \\ 0.503970540409 \\ -0.527187144651 \\ 0.378619988091 \\ 0.269255196301 \\ 0.671597043594 \\ 0.396010142274 \\ 0.325228576643 \\ 0.657808327574 \\ 0.801655023993 \\ 0.0907679484097 \\ -0.0713852594543 \\ -0.0691046005285 \\ -0.222582013343 \\ -0.848408031595 \\ -0.584285069026 \\ -0.324874831799 \\ 0.625621583197 \end{bmatrix}
$$

$U_{Z_1} \sim \text{Bernoulli}(0.352913861526), U_{Z_2} \sim \text{Bernoulli}(0.460995855543)$,
$U_{Z_3} \sim \text{Bernoulli}(0.331702473392), U_{Z_4} \sim \text{Bernoulli}(0.885505026779)$,
$U_{Z_5} \sim \text{Bernoulli}(0.017026872706), U_{Z_6} \sim \text{Bernoulli}(0.380772701708)$,
$U_{Z_7} \sim \text{Bernoulli}(0.028092602705), U_{Z_8} \sim \text{Bernoulli}(0.220819399962)$,
$U_{Z_9} \sim \text{Bernoulli}(0.617742227477), U_{Z_{10}} \sim \text{Bernoulli}(0.981975046713)$,
$U_{Z_{11}} \sim \text{Bernoulli}(0.142042291381), U_{Z_{12}} \sim \text{Bernoulli}(0.833602592350)$,
$U_{Z_{13}} \sim \text{Bernoulli}(0.882938907115), U_{Z_{14}} \sim \text{Bernoulli}(0.542143191999)$,
$U_{Z_{15}} \sim \text{Bernoulli}(0.085023436884), U_{Z_{16}} \sim \text{Bernoulli}(0.645357252864)$,
$U_{Z_{17}} \sim \text{Bernoulli}(0.863787135134), U_{Z_{18}} \sim \text{Bernoulli}(0.460539711624)$,
$U_{Z_{19}} \sim \text{Bernoulli}(0.314014079207), U_{Z_{20}} \sim \text{Bernoulli}(0.685879388218)$,
$U_X \sim \text{Bernoulli}(0.601680857267), U_Y \sim \text{Bernoulli}(0.497668975278)$,
$C_Y = -0.77953605542$.

A.1.2. COVARIATE

Covariate SCM is shown in figure 4b. The coefficients for Y_Z and C_Y were uniformly generated from the range $[-1, 1]$, while the parameters of the Bernoulli distribution were uniformly generated from $[0, 1]$. The detailed model is as follows:

$$
\begin{cases}
Z_i & = U_{Z_i} \text{ for } i \in \{1, ..., 20\}, \\
X & = f_X(U_X) \\
& = \begin{cases} 1 & \text{if } U_X > 0.5 \\ 0 & \text{otherwise,} \end{cases} \\
Y & = f_Y(X, Y_Z, U_Y) \\
& = \begin{cases} 1 & \text{if } 0 < C_Y \cdot X + Y_Z + U_Y < 1 \\ 1 & \text{if } 1 < C_Y \cdot X + Y_Z + U_Y < 2 \\ 0 & \text{otherwise.} \end{cases}
\end{cases}
$$

where, U_{Z_i}, U_X, U_Y are binary exogenous variables with Bernoulli distributions.
$s.t.$,

$$
Y_Z = \begin{bmatrix} Z_1 \; Z_2 \; ... \; Z_{20} \end{bmatrix} \times \begin{bmatrix} -0.792867111918 \\ 0.759967136147 \\ 0.55437722369 \\ 0.503970540409 \\ -0.527187144651 \\ 0.378619988091 \\ 0.269255196301 \\ 0.671597043594 \\ 0.396010142274 \\ 0.325228576643 \\ 0.657808327574 \\ 0.801655023993 \\ 0.0907679484097 \\ -0.0713852594543 \\ -0.0691046005285 \\ -0.222582013343 \\ -0.848408031595 \\ -0.584285069026 \\ -0.324874831799 \\ 0.625621583197 \end{bmatrix}
$$

$$U_{Z_1} \sim \text{Bernoulli}(0.352913861526), U_{Z_2} \sim \text{Bernoulli}(0.460995855543),$$
$$U_{Z_3} \sim \text{Bernoulli}(0.331702473392), U_{Z_4} \sim \text{Bernoulli}(0.885505026779),$$
$$U_{Z_5} \sim \text{Bernoulli}(0.017026872706), U_{Z_6} \sim \text{Bernoulli}(0.380772701708),$$
$$U_{Z_7} \sim \text{Bernoulli}(0.028092602705), U_{Z_8} \sim \text{Bernoulli}(0.220819399962),$$
$$U_{Z_9} \sim \text{Bernoulli}(0.617742227477), U_{Z_{10}} \sim \text{Bernoulli}(0.981975046713),$$
$$U_{Z_{11}} \sim \text{Bernoulli}(0.142042291381), U_{Z_{12}} \sim \text{Bernoulli}(0.833602592350),$$
$$U_{Z_{13}} \sim \text{Bernoulli}(0.882938907115), U_{Z_{14}} \sim \text{Bernoulli}(0.542143191999),$$
$$U_{Z_{15}} \sim \text{Bernoulli}(0.085023436884), U_{Z_{16}} \sim \text{Bernoulli}(0.645357252864),$$
$$U_{Z_{17}} \sim \text{Bernoulli}(0.863787135134), U_{Z_{18}} \sim \text{Bernoulli}(0.460539711624),$$
$$U_{Z_{19}} \sim \text{Bernoulli}(0.314014079207), U_{Z_{20}} \sim \text{Bernoulli}(0.685879388218),$$
$$U_X \sim \text{Bernoulli}(0.601680857267), U_Y \sim \text{Bernoulli}(0.497668975278),$$
$$C_Y = -0.77953605542.$$

A.1.3. DIRECT

Direct SCM is shown in figure 4c. The coefficient for C_Y were uniformly generated from the range $[-1, 1]$, while the parameters of the Bernoulli distribution were uniformly generated from $[0, 1]$. The detailed model is as follows:

$$\begin{cases} Z_i &= U_{Z_i} \text{ for } i \in \{1, ..., 20\}, \\ X &= f_X(U_X) \\ &= \begin{cases} 1 & \text{if } U_X > 0.5 \\ 0 & \text{otherwise,} \end{cases} \\ Y &= f_Y(X, U_Y) \\ &= \begin{cases} 1 & \text{if } C_Y \cdot X + U_Y > 0.7 \\ 0 & \text{otherwise.} \end{cases} \end{cases}$$

where, U_{Z_i}, U_X, U_Y are binary exogenous variables with Bernoulli distributions.

$$U_{Z_1} \sim \text{Bernoulli}(0.352913861526), U_{Z_2} \sim \text{Bernoulli}(0.460995855543),$$
$$U_{Z_3} \sim \text{Bernoulli}(0.331702473392), U_{Z_4} \sim \text{Bernoulli}(0.885505026779),$$
$$U_{Z_5} \sim \text{Bernoulli}(0.017026872706), U_{Z_6} \sim \text{Bernoulli}(0.380772701708),$$
$$U_{Z_7} \sim \text{Bernoulli}(0.028092602705), U_{Z_8} \sim \text{Bernoulli}(0.220819399962),$$
$$U_{Z_9} \sim \text{Bernoulli}(0.617742227477), U_{Z_{10}} \sim \text{Bernoulli}(0.981975046713),$$
$$U_{Z_{11}} \sim \text{Bernoulli}(0.142042291381), U_{Z_{12}} \sim \text{Bernoulli}(0.833602592350),$$
$$U_{Z_{13}} \sim \text{Bernoulli}(0.882938907115), U_{Z_{14}} \sim \text{Bernoulli}(0.542143191999),$$
$$U_{Z_{15}} \sim \text{Bernoulli}(0.085023436884), U_{Z_{16}} \sim \text{Bernoulli}(0.645357252864),$$
$$U_{Z_{17}} \sim \text{Bernoulli}(0.863787135134), U_{Z_{18}} \sim \text{Bernoulli}(0.460539711624),$$
$$U_{Z_{19}} \sim \text{Bernoulli}(0.314014079207), U_{Z_{20}} \sim \text{Bernoulli}(0.685879388218),$$
$$U_X \sim \text{Bernoulli}(0.601680857267), U_Y \sim \text{Bernoulli}(0.497668975278),$$
$$C_Y = -0.77953605542.$$

A.1.4. MEDIATOR

Mediator SCM is shown in figure 4a. The coefficients for X_Z, Y_Z, C_M, C_{Y_M}, and C_{Y_X} were uniformly generated from the range $[-1, 1]$, while the parameters of the Bernoulli distribution were uniformly generated from $[0, 1]$. The detailed model is

as follows:

$$
\begin{cases}
Z_i &= U_{Z_i} \text{ for } i \in \{1, ..., 20\}, \\
X &= f_X(X_Z, U_X) \\
&= \begin{cases} 1 & \text{if } X_Z + U_X > 0.5 \\ 0 & \text{otherwise,} \end{cases} \\
M &= f_M(X, U_M) \\
&= \begin{cases} 1 & \text{if } C_M \cdot X + U_M > 0.5 \\ 0 & \text{otherwise,} \end{cases} \\
Y &= f_Y(X, M, Y_Z, U_Y) \\
&= \begin{cases} 1 & \text{if } C_{Y_X} \cdot X + C_{Y_M} \cdot M + Y_Z + U_Y > 2 \\ 0 & \text{otherwise.} \end{cases}
\end{cases}
$$

where, U_{Z_i}, U_X, U_Y, U_M are binary exogenous variables with Bernoulli distributions.

$s.t.$,

$$
X_Z = \begin{bmatrix} Z_1 \; Z_2 \; ... \; Z_{20} \end{bmatrix} \times \begin{bmatrix} 0.259223510143 \\ -0.658140989167 \\ -0.75025831768 \\ 0.162906462426 \\ 0.652023463285 \\ -0.0892939586541 \\ 0.421469107769 \\ -0.443129684766 \\ 0.802624388789 \\ -0.225740978499 \\ 0.716621631717 \\ 0.0650682260309 \\ -0.220690334026 \\ 0.156355773665 \\ -0.50693672491 \\ -0.707060278115 \\ 0.418812816935 \\ -0.0822118703986 \\ 0.769299853833 \\ -0.511585391002 \end{bmatrix}, Y_Z = \begin{bmatrix} Z_1 \; Z_2 \; ... \; Z_{20} \end{bmatrix} \times \begin{bmatrix} -0.792867111918 \\ 0.759967136147 \\ 0.55437722369 \\ 0.503970540409 \\ -0.527187144651 \\ 0.378619988091 \\ 0.269255196301 \\ 0.671597043594 \\ 0.396010142274 \\ 0.325228576643 \\ 0.657808327574 \\ 0.801655023993 \\ 0.0907679484097 \\ -0.0713852594543 \\ -0.0691046005285 \\ -0.222582013343 \\ -0.848408031595 \\ -0.584285069026 \\ -0.324874831799 \\ 0.625621583197 \end{bmatrix}
$$

$$
\begin{aligned}
&U_{Z_1} \sim \text{Bernoulli}(0.352913861526), U_{Z_2} \sim \text{Bernoulli}(0.460995855543), \\
&U_{Z_3} \sim \text{Bernoulli}(0.331702473392), U_{Z_4} \sim \text{Bernoulli}(0.885505026779), \\
&U_{Z_5} \sim \text{Bernoulli}(0.017026872706), U_{Z_6} \sim \text{Bernoulli}(0.380772701708), \\
&U_{Z_7} \sim \text{Bernoulli}(0.028092602705), U_{Z_8} \sim \text{Bernoulli}(0.220819399962), \\
&U_{Z_9} \sim \text{Bernoulli}(0.617742227477), U_{Z_{10}} \sim \text{Bernoulli}(0.981975046713), \\
&U_{Z_{11}} \sim \text{Bernoulli}(0.142042291381), U_{Z_{12}} \sim \text{Bernoulli}(0.833602592350), \\
&U_{Z_{13}} \sim \text{Bernoulli}(0.882938907115), U_{Z_{14}} \sim \text{Bernoulli}(0.542143191999), \\
&U_{Z_{15}} \sim \text{Bernoulli}(0.085023436884), U_{Z_{16}} \sim \text{Bernoulli}(0.645357252864), \\
&U_{Z_{17}} \sim \text{Bernoulli}(0.863787135134), U_{Z_{18}} \sim \text{Bernoulli}(0.460539711624), \\
&U_{Z_{19}} \sim \text{Bernoulli}(0.314014079207), U_{Z_{20}} \sim \text{Bernoulli}(0.685879388218), \\
&U_X \sim \text{Bernoulli}(0.698319142733), U_Y \sim \text{Bernoulli}(0.502331024722), \\
&U_M \sim \text{Bernoulli}(0.402331024722), C_M = -0.74234511918. \\
&C_{Y_M} = 0.24235642321, C_{Y_X} = 0.87953605542.
\end{aligned}
$$

A.2. Detailed Data Generating Process

A.2.1. CONFOUNDER

Confounder SCM (Figure 4a). When all 20 binary features are observed, let $z = (z_1, \ldots, z_{20})$ and denote $X_Z(z)$ and $Y_Z(z)$ as fixed constants. The causal quantities are

$$\begin{aligned} PNS(z) &= P\big(Y = 0_{X=0},\, Y = 1_{X=1} \mid z\big) \\ &= P(U_Y = 0)\, T_0 + P(U_Y = 1)\, T_1, \\ T_0 &= \begin{cases} 1, & \text{if } f_Y(0, Y_Z(z), 0) = 0 \text{ and } f_Y(1, Y_Z(z), 0) = 1, \\ 0, & \text{otherwise,} \end{cases} \\ T_1 &= \begin{cases} 1, & \text{if } f_Y(0, Y_Z(z), 1) = 0 \text{ and } f_Y(1, Y_Z(z), 1) = 1, \\ 0, & \text{otherwise.} \end{cases} \end{aligned}$$

$$\begin{aligned} P\big(Y = 1 \mid do(X), z\big) &= \sum_{u_y} P(U_Y = u_y)\, f_Y\big(X, Y_Z(z), u_y\big), \\ P\big(Y = 1 \mid X, z\big) &= \sum_{u_x, u_y} P(U_X = u_x) P(U_Y = u_y)\, f_Y\big(f_X(X_Z(z), u_x), Y_Z(z), u_y\big). \end{aligned}$$

If only the first 10 features are observed, let $c = (z_1, \ldots, z_{10})$. Each subpopulation c contains $2^{10} = 1024$ full feature vectors $s_0, \ldots, s_{1023}$. Then

$$\begin{aligned} PNS_{\text{subpop}}(c) &= \sum_{j=0}^{1023} \frac{P(s_j)}{P(c)}\, PNS(s_j), \\ P\big(Y = 1 \mid do(X), c\big) &= \sum_{j=0}^{1023} \frac{P(s_j)}{P(c)}\, P\big(Y = 1 \mid do(X), s_j\big), \\ P\big(Y = 1 \mid X, c\big) &= \sum_{j=0}^{1023} \frac{P(s_j)}{P(c)}\, P\big(Y = 1 \mid X, s_j\big), \end{aligned}$$

where $P(s_j)$ is the product of the Bernoulli priors for $Z_{11:20}$.

A.2.2. COVARIATE

Outcome-Covariate SCM (Figure 4b). Here X is independent of Z; there is no X_Z. For any z:

$$\begin{aligned} PNS(z) &= P(U_Y = 0)\, T_0 + P(U_Y = 1)\, T_1, \quad \text{(definitions as above)}, \\ P\big(Y = 1 \mid do(X), z\big) &= \sum_{u_y} P(U_Y = u_y)\, f_Y\big(X, Y_Z(z), u_y\big), \\ P\big(Y = 1 \mid X, z\big) &= \sum_{u_x, u_y} P(U_X = u_x) P(U_Y = u_y)\, f_Y\big(f_X(u_x), Y_Z(z), u_y\big). \end{aligned}$$

Subpopulation-level formulas are identical to those in the Confounder case.

A.2.3. DIRECT

Direct SCM (Figure 4c). Here Z influences neither X nor Y; there is no Y_Z:

$$\begin{aligned} PNS(z) &= P(U_Y = 0)\, T_0 + P(U_Y = 1)\, T_1, \\ T_i &= \begin{cases} 1, & \text{if } f_Y(0,i) = 0 \text{ and } f_Y(1,i) = 1, \\ 0, & \text{otherwise,} \end{cases} \\ P\big(Y = 1 \mid do(X), z\big) &= \sum_{u_y} P(U_Y = u_y)\, f_Y(X, u_y), \\ P\big(Y = 1 \mid X, z\big) &= \sum_{u_x, u_y} P(U_X = u_x) P(U_Y = u_y)\, f_Y\big(f_X(u_x), u_y\big). \end{aligned}$$

Because Z is irrelevant, the weighted-average step over s_0:s_{1023} is omitted.

A.2.4. MEDIATOR

Mediator SCM (Figure 4d). For any full feature vector z:

$$\begin{aligned} X &= f_X\big(X_Z(z), U_X\big), \\ M &= f_M(X, U_M), \\ Y &= f_Y\big(X, M, Y_Z(z), U_Y\big). \end{aligned}$$

Individual-level quantities

$$\begin{aligned} PNS(z) &= \sum_{u_x, u_m, u_y} P(U_X) P(U_M) P(U_Y)\, \mathbf{1}\big[f_Y(0, f_M(0, u_m), Y_Z(z), u_y) = 0 \wedge \\ &\qquad f_Y(1, f_M(1, u_m), Y_Z(z), u_y) = 1\big], \\ P\big(Y = 1 \mid do(X), z\big) &= \sum_{u_m, u_y} P(U_M) P(U_Y)\, f_Y\big(X, f_M(X, u_m), Y_Z(z), u_y\big), \\ P\big(Y = 1 \mid X, z\big) &= \sum_{u_x, u_m, u_y} P(U_X) P(U_M) P(U_Y)\, f_Y\big(f_X(X_Z(z), u_x),\ f_M(f_X(X_Z(z), u_x), u_m), \\ &\qquad Y_Z(z), u_y\big). \end{aligned}$$

Subpopulation-level quantities With only Z_1:Z_{10} observed, define $c = (z_1, \ldots, z_{10})$ and enumerate s_0:s_{1023} as before. The three weighted-average formulas are identical to those in the Confounder subsection, with the individual-level expressions above substituted for $PNS(s_j)$, $P(Y = 1 \mid do(X), s_j)$, and $P(Y = 1 \mid X, s_j)$.

A.2.5. MASK-AUGMENTED QUERY CONSTRUCTION

We extend exact subpopulation queries $q \in \{0,1\}^d$ to mask-augmented queries $q \in \{0,1,X\}^d$, where $q_j = X$ indicates that the j-th covariate is unspecified (i.e., marginalized) in the query. Let $\mathcal{Z}(q) \subseteq \{0,1\}^d$ denote the set of exact assignments consistent with q (obtained by filling each X position with $\{0,1\}$).

Oracle aggregation (SCM). For any oracle quantity $g(z)$ defined on exact assignments (e.g., $P(y \mid do(x), z)$ or the oracle bound endpoints computed from SCM semantics), we define its value for a mask query q by mixture aggregation over consistent exact subpopulations:

$$\pi(q) = \sum_{z \in \mathcal{Z}(q)} P(z), \qquad g(q) = \frac{1}{\pi(q)} \sum_{z \in \mathcal{Z}(q)} P(z)\, g(z), \tag{7}$$

where $P(z)$ is the (SCM-implied) probability mass on exact z. We apply this aggregation to obtain oracle experimental and observational distributions conditioned on q, and then compute oracle PNS bounds for q by plugging the aggregated distributions into the Tian–Pearl formulas in the main text.

Sample-based bounds and thresholding. To emulate data scarcity, we generate finite observational and experimental samples and compute empirical frequencies for each exact $z \in \{0,1\}^d$. For a mask query q, its empirical distributions are obtained by aggregating counts over $\mathcal{Z}(q)$ (followed by normalization), mirroring the oracle mixture aggregation above. We then plug these empirical distributions into the Tian–Pearl formulas to obtain sample-based bounds $[LB(q), UB(q)]$ for training. Finally, we filter low-support queries using a threshold on the number of available samples (or equivalently, on the overlap/support statistics), and train models only on the retained queries.

A.3. Experiment Settings

The code is executed on the following hardware configuration:

- **CPU:** AMD Ryzen 7 5800H with Radeon Graphics @ 3.20 GHz
- **Memory:** 16 GB RAM
- **System Type:** 64-bit operating system
- **Architecture:** x64-based processor

The number of observational data and experimental data is from 50k to 500k.

A.3.1. Machine Learning Configuration

MLP backbone (Mish). All learning experiments use the same MLP backbone with Mish activations, and train *two* independent regressors: one for the lower bound and one for the upper bound.

- **Input representation:** each query $q \in \{0, 1, X\}^d$ is encoded by a $3d$-dimensional one-hot vector. For each coordinate $j \in [d]$, we use three channels indicating $q_j \in \{0, 1, X\}$, and concatenate them into $\phi(q) \in \{0, 1\}^{3d}$.
- **Architecture:** fully-connected MLP
$$3d \to 64 \to 32 \to 16 \to 1,$$
with Mish after each hidden layer and a final Sigmoid to ensure predictions lie in $[0, 1]$.
- **Outputs:** two separate models are trained, denoted $\widehat{LB}(q)$ and $\widehat{UB}(q)$.
- **Loss:** mean squared error (MSE) between predictions and test. (trained separately for LB and UB).
- **Optimiser:** Adam with learning rate 10^{-3}.
- **Train/validation split:** $90\%/10\%$ random split; we select the best checkpoint by validation MSE.
- **Training epochs:** up to 2000 epochs; validation is evaluated periodically and the best checkpoint is saved.
- **Batch size:** 4096.

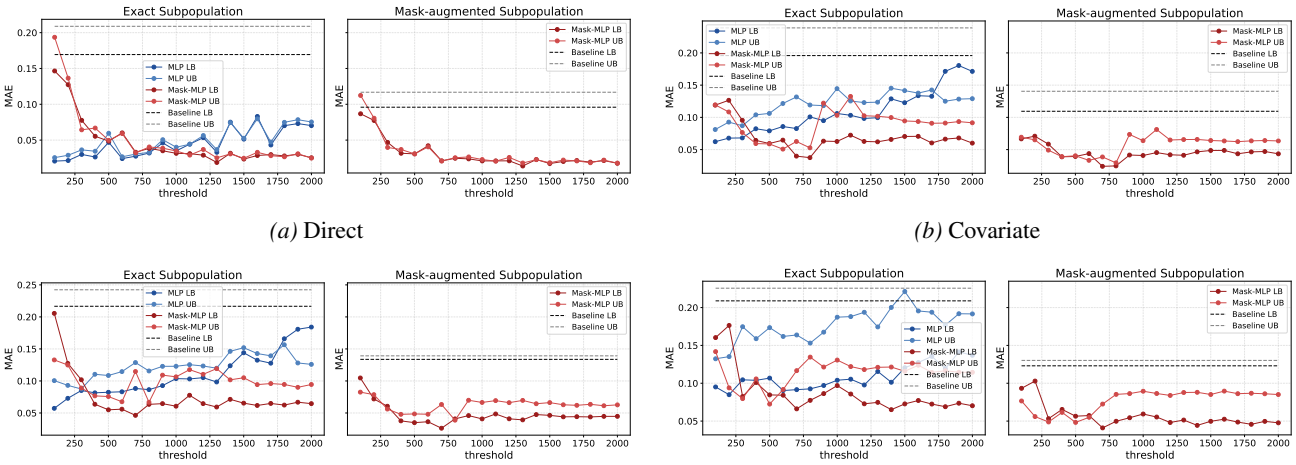
Training protocols We implement three evaluation settings, using the same MLP backbone and training data construction as in the main text:

- **MLP for Exact Subpopulation (Exact-MLP).** Train on thresholded *exact* queries only, i.e., $q \in \{0, 1\}^d$ (no masked entries). Evaluate on the oracle bounds of the exact-query subset.
- **Mask-MLP for Exact Subpopulation.** Train a single Mask-MLP on the thresholded *mask-augmented* supervision space, i.e., $q \in \{0, 1, X\}^d$. Evaluate on the oracle bounds of the exact-query subset $q \in \{0, 1\}^d$, enabling direct comparison with MLP for Exact Subpopulation.
- **Mask-MLP for Mask-augmented Subpopulation.** Using the same Mask-MLP trained above, evaluate on the oracle bounds over the full mask-augmented query space $q \in \{0, 1, X\}^d$.

A.4. Experimental Result

This appendix reports additional diagnostic plots for all budgets. For each budget, we include (i) threshold sweeps of MAE (each PDF contains two panels: exact queries on the left and mask-augmented queries on the right), and (ii) qualitative true-vs-predicted plots on oracle bounds.

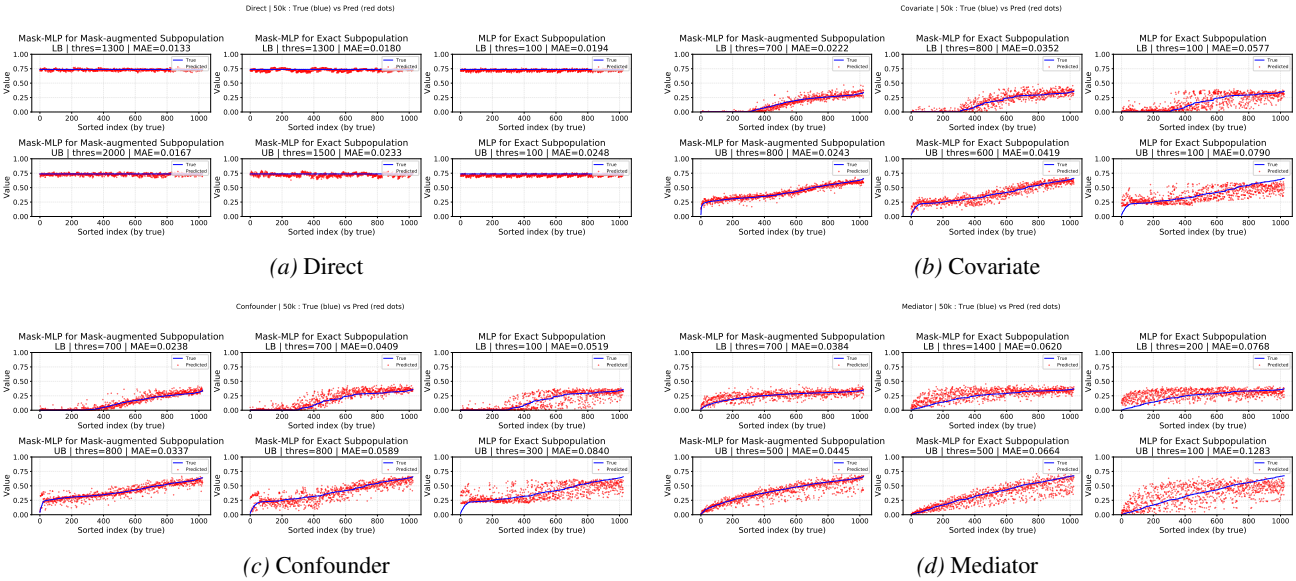
A.4.1. BUDGET = 50K



(a) Direct *(b)* Covariate

(c) Confounder *(d)* Mediator

Figure 5. Threshold sweeps (50k budget). Each PDF contains two panels: evaluation on exact queries (left) and on mask-augmented queries (right).

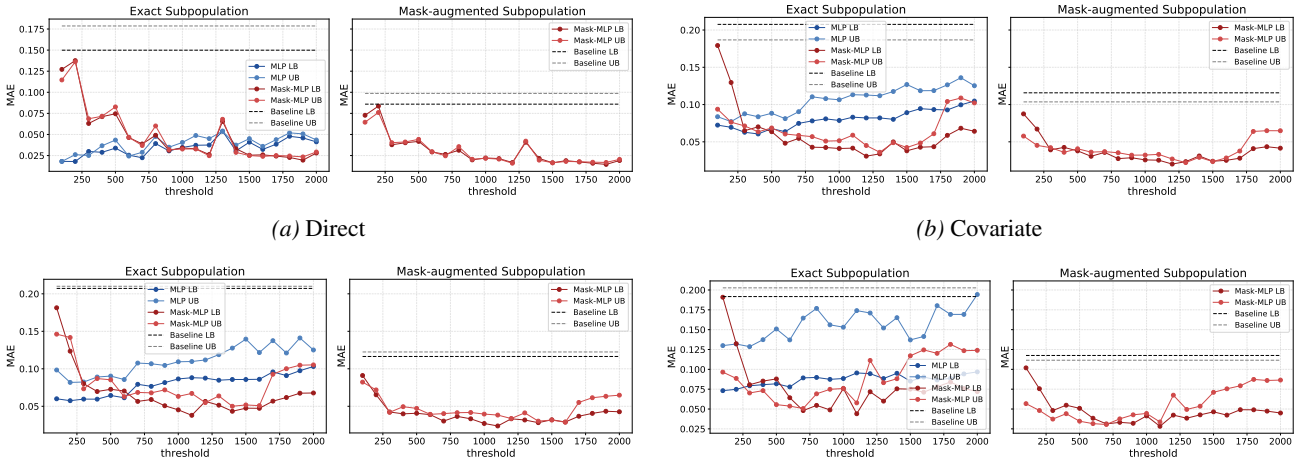


(a) Direct *(b)* Covariate

(c) Confounder *(d)* Mediator

Figure 6. True vs. predicted oracle bounds (50k budget). Each PDF visualizes prediction fidelity; see legends within each plot for exact vs. mask-augmented evaluations.

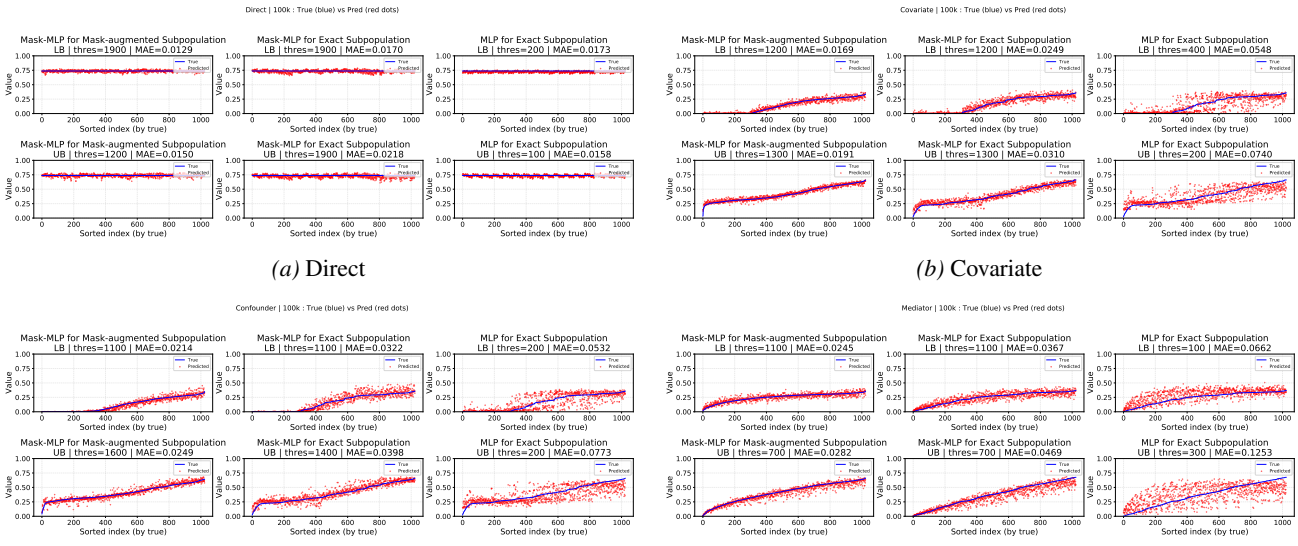
A.4.2. BUDGET = 100K



(a) Direct *(b)* Covariate

(c) Confounder *(d)* Mediator

Figure 7. Threshold sweeps (100k budget). Each PDF contains two panels: evaluation on exact queries (left) and on mask-augmented queries (right).

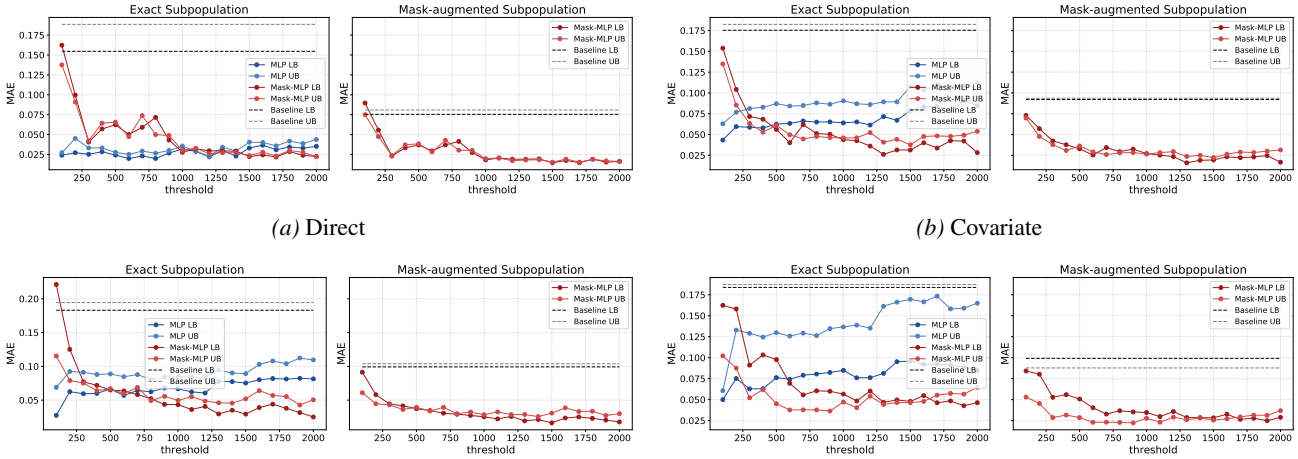


(a) Direct *(b)* Covariate

(c) Confounder *(d)* Mediator

Figure 8. True vs. predicted oracle bounds (100k budget).

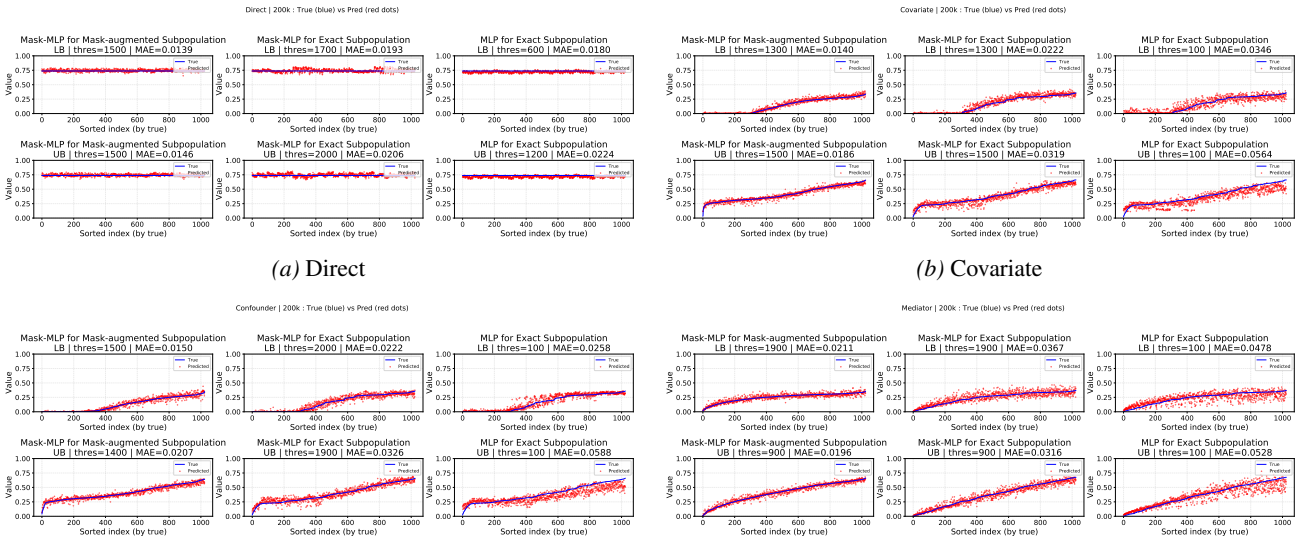
A.4.3. BUDGET = 200K



(a) Direct *(b)* Covariate

(c) Confounder *(d)* Mediator

Figure 9. Threshold sweeps (200k budget). Each PDF contains two panels: evaluation on exact queries (left) and on mask-augmented queries (right).

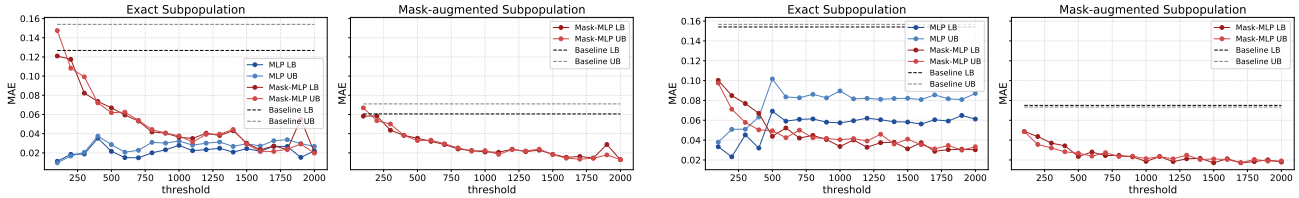


(a) Direct *(b)* Covariate

(c) Confounder *(d)* Mediator

Figure 10. True vs. predicted oracle bounds (200k budget).

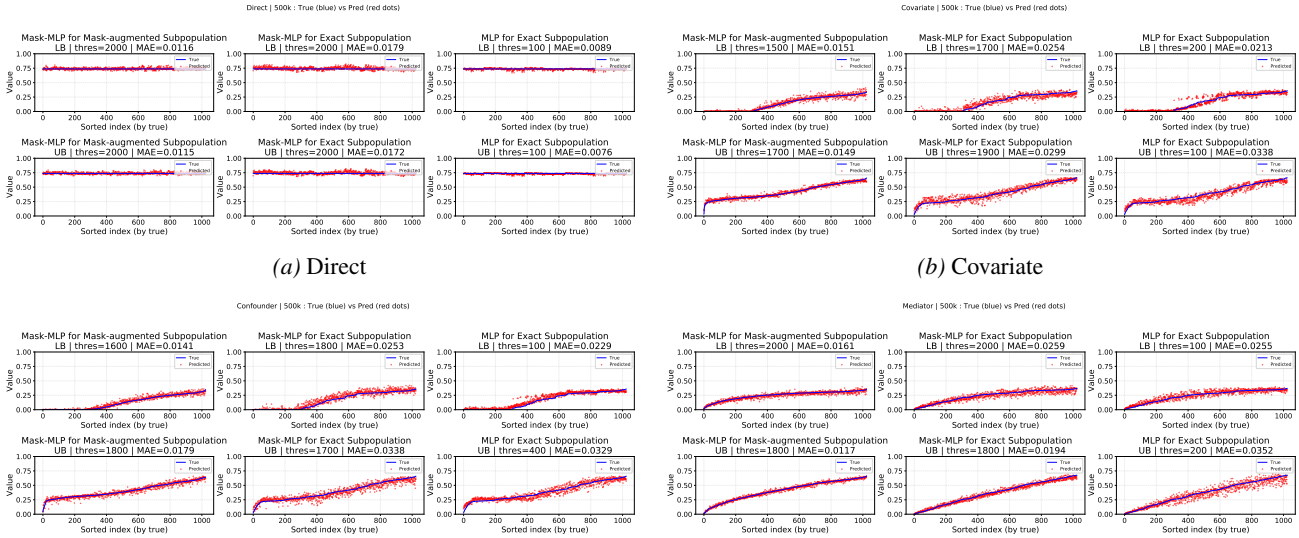
A.4.4. BUDGET = 500K



(a) Direct *(b)* Covariate

(c) Confounder *(d)* Mediator

Figure 11. Threshold sweeps (500k budget). Each PDF contains two panels: evaluation on exact queries (left) and on mask-augmented queries (right).



(a) Direct *(b)* Covariate

(c) Confounder *(d)* Mediator

Figure 12. True vs. predicted oracle bounds (500k budget).