A CAUSAL FOUNDATION MODEL FOR STRUCTURE AND OUTCOME PREDICTION

Anonymous authors
Paper under double-blind review

ABSTRACT

We introduce TabPFN-CFM, a causal foundation model that addresses causal structure learning and causal outcome estimation within a single amortized inference network. TabPFN-CFM is a prior-fitted network trained purely on synthetic structural causal models. Using only an observational dataset and, optionally, prior knowledge of the causal graph, it predicts the posterior causal structure, including unobserved confounding, and distributions for observational, interventional and counterfactual queries, covering all three levels of Pearl's Causal Hierarchy. Representing structure with acyclic directed mixed graphs lets the model express latent confounding, and we show that conditioning on a known graph can only improve predictive accuracy. Evaluated on in-distribution, out-of-distribution and semi-synthetic datasets with established causal graphs, TabPFN-CFM improves over meta-learner and existing prior-fitted-network baselines for interventional prediction and over both classical and deep structure learners for graph recovery, while also supporting counterfactual queries. A set of backbone and optimizer changes reduces the number of training steps needed to reach a target loss by roughly $4\times$.

1 INTRODUCTION

Answering a causal question requires more than fitting a predictive model to observational data: it requires knowing which variables cause which, whether unobserved factors confound the relationship of interest, and how the system would have behaved under a different action. Recovering causal structure and causal outcomes from observational data alone is a long-standing problem (Pearl, 2000; Imbens & Rubin, 2015), and classical practice splits it into two separate pipelines. Causal discovery methods such as PC, FCI, GES and LiNGAM (Spirtes et al., 2000; 1995; Chickering, 2002; Shimizu et al., 2006) recover a graph, or an equivalence class of graphs. Causal effect estimators such as meta-learners (Künzel et al., 2019) then assume a graph, or at least a valid adjustment set, is already given. Errors in the first stage propagate into the second, and neither stage produces calibrated predictive distributions.

Amortized inference offers an alternative for both tasks. Prior-fitted networks (PFNs) (Hollmann et al., 2023) are trained on data sampled from a synthetic prior and learn to approximate Bayesian posterior prediction in a single forward pass. Do-PFN (Robertson et al., 2025) and CausalPFN (Balazadeh et al., 2025) apply this idea to interventional outcome estimation, while AVICI (Lorch et al., 2022) and CSIvA (Ke et al., 2023) apply it to structure learning. Each solves one half of the problem. A model that recovers structure cannot answer a query about an intervention; a model that answers interventional queries treats the graph as a nuisance and so cannot report it, cannot exploit it when a practitioner knows part of it, and cannot flag that a query is confounded.

We argue that these tasks should be solved jointly. Structure and outcome are two readouts of the same latent object — the structural causal model (SCM) that generated the data — so a network that amortizes the posterior over SCMs can produce both. Joint training also acts as a mutual regularizer: predicting interventional outcomes forces the representation to encode causal direction, and predicting structure forces it to encode which variables are confounded. We confirm this: removing the structure loss degrades interventional and counterfactual accuracy while leaving observational accuracy unchanged (Section 6.4). We introduce **TabPFN-CFM** (TabPFN-Causal Foundation Model), a causal foundation model trained entirely on synthetic SCMs. Our contributions are as follows.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

- **A single model for structure and outcome.** TabPFN-CFM predicts an acyclic directed mixed graph (ADMG) alongside predictive distributions for outcome queries, in one forward pass.
- **All three rungs of Pearl's Causal Hierarchy.** One network answers observational, interventional and counterfactual queries, selected by a query-type token. To our knowledge this is the first PFN to support counterfactual queries.
- **Optional graph conditioning.** A known graph can be supplied to sharpen the predictive distribution. We show this can never increase the expected loss, and strictly helps whenever the graph changes the posterior over SCMs (Theorem 2).
- **Explicit modeling of unobserved confounding.** Unlike deep structure learners that output a DAG over observed variables, we predict an ADMG and so report which pairs of variables share a hidden common cause.
- **An efficient training recipe.** Backbone and optimizer changes reach a target prediction loss roughly $4\times$ faster and a target structure loss roughly $3\times$ faster at matched parameter count (Section 6.4).

2 Related Work

Causal structure learning. Constraint-based methods such as PC (Spirtes et al., 2000) and FCI (Spirtes et al., 1995) recover a graph from conditional-independence tests, with FCI able to signal latent confounding; score-based methods such as GES (Chickering, 2002) search over equivalence classes; and functional-form methods such as LiNGAM (Shimizu et al., 2006) exploit non-Gaussianity to orient Markov-equivalent edges. These methods return a graph rather than a calibrated posterior, and their guarantees rest on assumptions of faithfulness, linearity, and causal sufficiency, which are hard to verify. Amortized alternatives train a network on simulated graphs: AVICI (Lorch et al., 2022) performs variational inference over adjacency matrices, and CSIvA (Ke et al., 2023) trains a transformer to emit the adjacency matrix directly. Arrow (Thompson et al., 2026) enforces acyclicity in the learned graph. All three output a DAG over the observed variables and therefore cannot express unobserved confounding. While CSIvA uses an auxiliary loss over immediate parents and children to regularize training, our decoder explicitly emits an ancestral matrix together with the ADMG. We find this direct ancestral prediction is more accurate than deriving ancestry from a sampled adjacency matrix (Table 10).

Causal effect estimation. Given a graph or a valid adjustment set, meta-learners reduce effect estimation to supervised learning: the S-, T- and X-learners (Künzel et al., 2019) and the doubly robust learner (Battocchi et al., 2019) wrap an arbitrary regressor to produce conditional average treatment effects. Their accuracy is inherited from the base learner and their validity from the assumed adjustment set; they do not estimate structure and, being point estimators of a contrast between potential outcomes, cannot answer counterfactual queries about a specific unit.

Prior-fitted networks and latent confounding. PFNs (Hollmann et al., 2023) are transformers trained on datasets sampled from a synthetic prior to approximate the Bayesian posterior predictive distribution in a single forward pass. Do-PFN (Robertson et al., 2025) extends the recipe to interventional queries by training on SCM-derived priors, and CausalPFN (Balazadeh et al., 2025) asserts strong ignorability in the prior to ensure identifiability. We build on this line of work, extending the prior with latent variables and counterfactual targets, the architecture with a graph encoder and structure decoder, and the query set to the third rung of the causal hierarchy. Marginalizing latent variables out of a DAG yields an acyclic directed mixed graph (Richardson, 2003; Zhao, 2025), in which a bidirected edge records a hidden common cause; ADMGs are standard in the identification literature but are rarely the output of a learned model. Recently, Reuter et al. (2026) has explored including ancestral graph information through attention biasing in PFNs.

3 Background and Problem Setup

3.1 Structural causal models and mixed graphs

We assume the system follows an SCM $\psi = \{U, V, F\}$ with unobserved variables U (latent covariates and sources of exogenous noise), observed variables V split into covariates X, a binary treatment T and an outcome Y, and unknown structural equations F. Standard treatments assume a *nonparametric structural equation model with independent errors* (NPSEM-IE) (Pearl, 2000), under which the causal graph over V is a DAG; we do not, so latents may be endogenous, may have several children in V, and may be dependent. Marginalizing U out then yields an *acyclic directed mixed graph* (ADMG) (Richardson, 2003; Zhao, 2025), written $\mathcal{G} = \{A, C\}$, with adjacency matrix $A \in \{0,1\}^{(k+2)\times(k+2)}$ recording causal influence and symmetric matrix C recording shared unobserved causes, over the k covariates plus treatment and outcome. Latent confounding is common in practice and is the regime in which the graph may fail to be identified from observational data, so reporting C both describes the structure and signals how much to trust the associated effect estimate. The NPSEM-IE conditions and a fuller treatment of ADMGs are given in Section A.

3.2 Causal queries

We are given an observational dataset $\mathcal{D}^{\text{obs}} = \{\mathbf{x}_i, y_i, t_i\}_{i=1}^n$ of n i.i.d. samples drawn from ψ and, optionally, prior knowledge of the causal graph $\mathcal{G}^{\text{est}}$; when no prior is available $\mathcal{G}^{\text{est}} = \varnothing$. The model targets two objectives.

The first is **structure prediction**: when the true causal graph is unknown, estimate the ADMG underlying the observed variables. The second is **outcome prediction** under the three levels of Pearl's Causal Hierarchy (Pearl, 2000). Given a held-out unit with covariates $\mathbf{x}^*$, natural treatment t^* and factual outcome y^*, and writing $\mathcal{C} = (\mathcal{D}^{\text{obs}}, \mathcal{G}^{\text{est}})$ for the conditioning set, the three queries are

$$\begin{aligned} \text{Observational:} \quad & P(y^* \mid \mathbf{x}^*, T = t^*, \mathcal{C}) \\ \text{Interventional:} \quad & P(y^* \mid \mathbf{x}^*, \text{do}(T = 1 - t^*), \mathcal{C}) \\ \text{Counterfactual:} \quad & P(y^* \mid \mathbf{x}^*, \text{do}(T = 1 - t^*), y = y_{t^*}, \mathcal{C}) \end{aligned}$$

The interventional query assigns a treatment externally, while the counterfactual query conditions in addition on the outcome actually observed under the original treatment, and is therefore more informative for that unit. Training a single model on all three tasks gives it a broader view of causal processes and, because the representation is shared, is more sample-efficient than training a separate model per task.

3.3 Amortized Bayesian inference over SCMs

We follow the Bayesian PFN framework (Hollmann et al., 2023; Robertson et al., 2025; Balazadeh et al., 2025). A prior over SCMs, $p(\psi)$, generates the true SCM $\psi_{\text{true}} \sim p(\psi)$; observational data then induces the posterior $P(\psi \mid \mathcal{D}^{\text{obs}}, \mathcal{G}^{\text{est}})$. Given an SCM, an outcome is determined by $P(y \mid \mathbf{x}^*, \text{do}(T = t^*), \psi)$, so the target distribution for, say, an interventional query is

$$P(y^* \mid \mathbf{x}^*, \text{do}(T = 1 - t^*), \mathcal{D}^{\text{obs}}, \mathcal{G}^{\text{est}}) = \int P(y^* \mid \mathbf{x}^*, \text{do}(T = 1 - t^*), \psi)\, P(\psi \mid \mathcal{D}^{\text{obs}}, \mathcal{G}^{\text{est}})\, d\psi, \tag{1}$$

and, when the graph is unknown, its posterior is $P(\mathcal{G}^* \mid \mathcal{D}^{\text{obs}}) = \int P(\mathcal{G}^* \mid \psi)\, P(\psi \mid \mathcal{D}^{\text{obs}})\, d\psi$.

In practice we never form $P(\psi \mid \mathcal{D}^{\text{obs}})$ explicitly. Instead we draw SCMs from the prior and train a network $\hat{p}_\theta$ to estimate Equation (1) directly by minimizing the conditional negative log likelihood

$$L = -\mathbb{E}\left[\log \hat{p}_\theta\left(y^* \mid \mathbf{x}^*, \text{do}(T = 1 - t^*), \mathcal{D}^{\text{obs}}, \mathcal{G}\right)\right]. \tag{2}$$

Proposition 1 (Loss equivalence)**.** *Minimizing Equation* (2) *is equivalent to minimizing the expected KL divergence between the true posterior predictive distribution and the model's distribution,* $L = \mathbb{E}\big[\text{KL}\big(p(y \mid c, \mathcal{D}^{\text{obs}}, \mathcal{G}) \,\|\, \hat{p}_\theta(y \mid c, \mathcal{D}^{\text{obs}}, \mathcal{G})\big)\big] + C$*, where* c *collects the query inputs and* C *does not depend on* θ*.*

Proposition 2 (Graph conditioning cannot hurt)**.** *Let* L_0 *and* L_G *denote the expected negative log likelihood of the posterior predictive distributions computed without and with* $\mathcal{G}$ *as an input.*

Then $L_G - L_0 = -\mathbb{E}\big[\mathrm{KL}\big(p_G(y \mid \mathcal{D}^{\mathrm{obs}}, \mathcal{G}) \,\|\, p_0(y \mid \mathcal{D}^{\mathrm{obs}})\big)\big] \leq 0$*, with equality if and only if* $p(\psi \mid \mathcal{D}^{\mathrm{obs}}, \mathcal{G}) = p(\psi \mid \mathcal{D}^{\mathrm{obs}})$ *wherever* y *has support.*

Theorem 1 extends the corresponding results for TabPFN (Hollmann et al., 2023) and Do-PFN (Robertson et al., 2025) to the graph-conditioned case, and Theorem 2 justifies supplying a known graph as an input rather than ignoring it; both proofs are in Sections B and C. The degenerate case in Theorem 2 is the one in which the observational data already identifies the SCM, so the graph carries no additional information. In the finite-data regime this rarely holds even for identifiable SEMs, which is consistent with the gains we observe from graph conditioning in Section 6.

4 The Synthetic SCM Prior

Because TabPFN-CFM approximates Bayesian inference under a prior $p(\psi)$, the prior *is* the training set: everything the model can express about causal systems is inherited from the generative process. We extend the priors used in Do-PFN (Robertson et al., 2025) with latent variables, a broader family of nonlinearities and noise models, and counterfactual targets.

To draw an SCM we sample a random DAG, designate a treatment and outcome node — biased towards a treatment that is a causal ancestor of the outcome, so that interventions are meaningful — and sample a structural equation independently for every node, combining a nonlinearity (Table 5) with noise whose type, position and distribution are also drawn at random (Table 6). A random subset of nodes is then designated observed (V), with the remainder latent (U) and the treatment and outcome always observed; the probability that a node is latent is itself sampled per SCM, up to a maximum of 0.5 during training. Marginalizing out U produces the adjacency matrix A and the bidirected confounding matrix C that together form the ADMG. The full parameterization, including the binary treatment construction and the sampled distribution for every hyperparameter, is given in Section D.

One training example consists of a fit set $\mathcal{D}^{\mathrm{fit}} = \{\mathbf{x}_i, t_i, y_i\}_{i=1}^{n}$, obtained by evaluating the structural equations in topological order, a prediction query $\mathcal{D}^{\mathrm{pred}} = \{\mathbf{x}^*, t^*, y^*\}$, and the matched counterfactual $\mathcal{D}^{\mathrm{causal}} = \{\mathbf{x}_t^*,\, t = \mathrm{do}(1 - t^*),\, y_t^*\}$. Generating the counterfactual requires care: nodes lying between T and Y, whether observed or latent, may be descendants of T and must *not* be held fixed. We pin all noise variables and all non-descendants of T to their observed values, set $T = \mathrm{do}(1 - t^*)$, and recompute the descendants. Mediators therefore change under the intervention while staying tied to their factual values through the shared noise, which is the signal the model must learn to exploit.

The diversity of this prior is what allows the model to generalize: it never sees a real dataset during training, yet must stay accurate across graph families, functional forms, noise models and confounding patterns. Our model is trained with up to $n = 1600$ fit samples and $k = 33$ covariates (35 nodes in total).

5 Model Architecture

TabPFN-CFM extends the Do-PFN architecture with explicit graph conditioning and a structure decoder. It has four components: an *encoder*, an *observation module*, a *graph module* and a *prediction module* (Figure 1). We summarize each below; full equations and shapes are given in Section E.

5.1 Encoder

Write $\mathbf{x}^{\mathrm{fit}} \in \mathbb{R}^{n \times k}$, $\mathbf{x}^* \in \mathbb{R}^{1 \times k}$, $t^{\mathrm{fit}} \in \{0,1\}^n$, $y^{\mathrm{fit}} \in \mathbb{R}^n$ and $\mathcal{G} \in \{0,1\}^{(k+2)\times(k+2)}$, with hidden dimension h. Covariates are embedded with a shared linear map applied to the value concatenated with its empirical CDF, which stabilizes extreme outliers, followed by a column positional embedding drawn as $W_{\mathrm{col}}\,\epsilon$ with $\epsilon \sim \mathcal{N}(0, I_{h/4})$ and resampled on every forward pass, at both training and inference time, so the model cannot rely on a fixed column ordering. Outcomes are embedded from the value, its CDF and a type indicator distinguishing fit outcomes from counterfactual queries with and without an observed factual outcome. Treatments use a lookup table; the query treatment embedding has four states, encoding both its value and whether the query is observational or interventional.

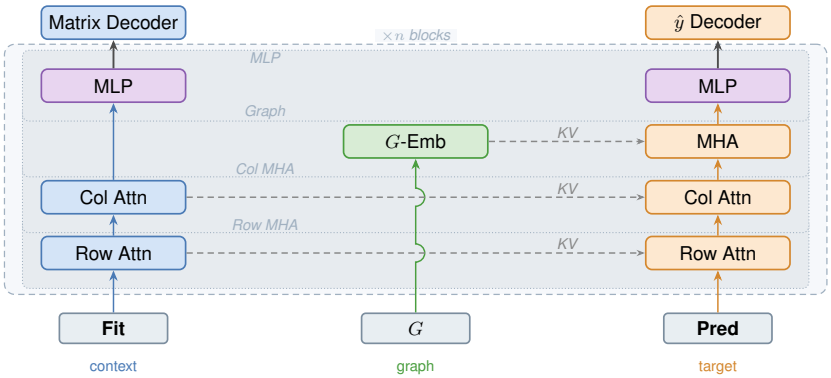

Figure 1: TabPFN-CFM architecture. The fit set, the query and the prior ADMG are embedded separately. The observation module processes the fit set with alternating row and column attention; the graph module fuses the ADMG embedding with intermediate fit-set states; the prediction module cross-attends to both. Structure is decoded from the fit-set states and therefore never depends on the query.

The prior ADMG is encoded from A and C, from which we derive the ancestral (transitive closure) matrix R; when the graph is unknown all matrices are zero. The embedding sums learned elementwise binary embeddings of A, R, C and their transposes, $h^G = \sum_{M\in\{A,R,C\}} \big(\mathrm{emb}_M(M)+\mathrm{emb}_{M^\top}(M^\top)\big)$. This representation is deliberately redundant — transposes and closures are functions of A and C — but it gives the model single-hop access to each node's parents, children, ancestors and descendants (Ke et al., 2023), which we find matters more than parameter count.

5.2 Observation, graph and prediction modules

The fit-set embeddings pass through L pre-layer-norm tabular transformer blocks, each comprising row-wise attention (across samples), column-wise attention (across features) and a feed-forward layer. The structure decoder averages the final feature states over fit rows, projects them through two linear layers with RMS normalization into source and destination embeddings u_i, v_i, and scores every candidate edge as $\hat{M}_{i,j} = \alpha \cdot \mathrm{gelu}(u_i^\top W) \cdot v_j + \beta$ for $\hat{M} \in \{\hat{A}, \hat{R}, \hat{C}\}$, with separate parameters per output matrix. This nonlinear decoder outperforms the plain dot product used in AVICI (Lorch et al., 2022). The structure prediction depends only on $\mathcal{D}^{\mathrm{fit}}$, never on the query or the supplied graph, so a user cannot bias the recovered structure through the query and the input graph cannot simply be copied to the output.

The graph module combines h^G with intermediate observation states: column-wise states are averaged over samples, mapped through separate source and destination projections, and combined by an outer product into H_l^{feat}, giving $H_l^G = \mathrm{LN}(h^G + H_l^{\mathrm{feat}})$. We keep h^G fixed at its encoder value across layers, as updating it gave no benefit. The prediction module mirrors the observation module but adds cross-attention over both the observation states and the graph states, so the query attends to the observational data and to the graph prior while attention never flows back from $\mathcal{D}^{\mathrm{pred}}$ to $\mathcal{D}^{\mathrm{fit}}$. As in TabPFN this allows efficient inference from a KV cache, and it is what makes the structure prediction query-independent. The outcome distribution is read off the final state of the outcome feature through an MLP head producing logits over b discretized buckets.

Relative to TabPFNv2 and Do-PFN we omit feature grouping, ensembling, random augmentations and random feature products, because the graph indexes variables explicitly and would not survive feature mixing. The CDF augmentation is retained as a fixed part of the encoder.

5.3 Training procedure

Each training datapoint consists of $\mathcal{D}^{\mathrm{fit}}$, $\mathcal{D}^{\mathrm{pred}} = \{\mathbf{x}^*, t^*, y^*\}$, $\mathcal{D}^{\mathrm{causal}} = \{\mathbf{x}_t^*, t = \mathrm{do}(1-t^*), y_t^*\}$ and $\mathcal{G}^{\mathrm{est}}$, and is used to generate one sample of each query type. For the observational query the model receives $\mathbf{x}^*$ and t^* and must predict y^*; for the interventional query it receives $\mathbf{x}^*$ and either $\mathrm{do}(1-t^*)$ or $\mathrm{do}(t^*)$ and must predict y_t^* or $y_{1-t^*}^*$, respectively; for the counterfactual query it additionally

receives the factual outcome y^* alongside $\mathbf{x}_t^*$ and $\mathrm{do}(1-t^*)$, and must predict y_t^*. In each case the prediction loss is the cross entropy over discretized outcome buckets, $\mathcal{L}_{\mathrm{pred}} = \mathrm{CE}(y_{\mathrm{targ}}, \hat{p}_\theta(y))$, the standard PFN objective (Hollmann et al., 2023; Balazadeh et al., 2025; Robertson et al., 2025). The fit set is always provided, whereas $\mathcal{G}^{\mathrm{est}}$ is zeroed out half the time so that a single set of weights serves both the graph-conditioned and graph-free regimes.

The structure loss is the elementwise binary cross entropy between predictions and true matrices, $\mathcal{L}_{\mathrm{graph}} = \frac{1}{(k+2)^2}\sum_{i,j}\mathrm{BCE}(M_{i,j}, \hat{M}_{i,j})$ summed over $(M, \hat{M}) \in \{(A, \hat{A}), (R, \hat{R}), (C, \hat{C})\}$, computed once per datapoint since structure predictions are query-independent. The overall objective is $\mathcal{L} = \mathcal{L}_{\mathrm{pred}} + \lambda\,\mathcal{L}_{\mathrm{graph}}$. Finally, the model is trained with dummy features arising from batch padding. Although samples are processed independently, the model can attend to these dummy features; keeping them at inference improves performance, suggesting they act as scratch space for intermediate computation.

5.4 Backbone and optimizer changes

Independently of the causal machinery above, we make several changes to the TabPFNv2 transformer backbone and training recipe, largely inspired by the Modded NanoGPT project (Jordan et al., 2024a). These add no new functionality; they target stability and throughput. The changes are: full attention to h_{row} in the prediction module; a cosine learning-rate schedule with 1000 linear warmup steps; relu^2 activations (Zhang et al., 2024); a reduced initialization scale for positional embeddings; pre-layer norm in place of post-layer norm (Xiong et al., 2020); a layer norm before the prediction MLP head; the Muon optimizer for matrix parameters (Jordan et al., 2024b) in the Kimi variant (Liu et al., 2025); QK norm with a learnable temperature (Henry et al., 2020); and an increase in width from 192 to 288, taking the model from 10.28M to 23.2M parameters. Their combined effect is quantified in Section 6.4.

6 Experiments

We evaluate TabPFN-CFM in four settings of increasing realism: analytically tractable toy SEMs, where predictions can be compared against exact posteriors (Section 6.1); large samples of synthetic SEMs drawn from the training prior, and out-of-distribution SEMs built from a family of nonlinearities never seen during training (Section 6.2); and real datasets with established causal graphs (Section 6.3). Section 6.4 then ablates the structure loss and isolates the contribution of the backbone changes.

For outcome prediction we compare against Do-PFN (Robertson et al., 2025), CausalPFN (Balazadeh et al., 2025) and the established meta-learners — S-learner, T-learner, X-learner and doubly robust (DR) learner — all using TabPFNv2.5 as the base model (Künzel et al., 2019; Battocchi et al., 2019); none of these baselines can express a counterfactual query. For structure prediction we compare against the amortized models AVICI (Lorch et al., 2022) and Arrow (Thompson et al., 2026), and the classical methods FCI (Spirtes et al., 1995), GES (Chickering, 2002), LiNGAM (Shimizu et al., 2006) and PC (Spirtes et al., 2000), of which only FCI can report confounding. We report our model both with and without the true graph supplied. Outcome prediction is scored by mean squared error (MSE) of the predictive mean; structure prediction by elementwise accuracy and area under the ROC curve (AUROC) against the true adjacency, ancestral and confounding matrices.

6.1 Toy SEMs with tractable posteriors

A linear instrumental variable SEM ($Z \to T \to Y$, $U \to T$, $U \to Y$) with known parameters admits exact distributions for all three query types, so predictions can be checked against ground truth. They match the exact posteriors well (Figures 3 and 4), including the counterfactual point mass. The interventional distribution is recovered even when U is hidden, and the predicted adjacency matrix (Table 9) matches the true structure. On a nonlinear SEM with a latent variable and mediators downstream of T, where the exact posterior is intractable, predictions again align with samples from the true distribution and improve when the graph is supplied; the counterfactual is biased towards the factual outcome y^* at small n and this bias vanishes as the posterior concentrates (Figure 6). Full details for both experiments are given in Sections F.1 and F.2.

Table 1: Outcome prediction on synthetic SEMs with $n = 1024$ fit samples, by mean squared error (↓). Baselines cannot express counterfactual queries. Best in bold.

	In distribution			Out of distribution		
Model	Obs.	Int.	Count.	Obs.	Int.	Count.
S-learner	**0.430**	0.797	-	0.565	0.888	-
T-/X-learner	0.434	0.846	-	0.574	0.909	-
DR-learner	0.431	0.825	-	0.577	0.885	-
Do-PFN	0.670	0.941	-	0.782	0.966	-
CausalPFN	0.436	0.777	-	0.613	0.903	-
TabPFN-CFM	0.432	0.576	0.246	0.534	0.755	0.603
TabPFN-CFM w/ graph	0.432	**0.550**	**0.216**	**0.534**	**0.734**	**0.575**

Table 2: Structure prediction on synthetic SEMs with $n = 1024$ fit samples, by AUROC (↑). Only FCI and TabPFN-CFM report confounding; accuracies are in Tables 12 and 14.

	In distribution			Out of distribution		
Model	Adj.	Anc.	Conf.	Adj.	Anc.	Conf.
FCI	0.533	0.536	0.576	0.535	0.504	0.560
GES	0.672	0.657	-	0.654	0.654	-
LiNGAM	0.568	0.540	-	0.558	0.524	-
PC	0.651	0.632	-	0.685	0.639	-
AVICI	0.703	0.682	-	0.623	0.596	-
Arrow	0.725	0.715	-	0.734	0.736	-
TabPFN-CFM	**0.898**	**0.916**	**0.868**	**0.880**	**0.894**	**0.804**

6.2 Synthetic and out-of-distribution SEMs

We next evaluate on large samples of synthetic SEMs, with 500 datasets drawn and cached for each evaluation setting. In-distribution SEMs come from the training prior of Section 4. The out-of-distribution (OOD) SEMs replace the neural-network nonlinearities with random Fourier features and raise the maximum probability that a node is unobserved from 0.5 to 0.75 (Section G), giving sparser observed graphs with more latent confounding. The Fourier family is used in other works but not in our training prior, so this tests our model's generalization.

Table 1 summarizes outcome prediction. Three findings stand out. First, in the observational setting TabPFN-CFM is comparable to the baselines, which is expected: the meta-learners are built on the strong TabPFNv2.5 base model, and observational prediction is an associative task where causal knowledge does not help. Second, TabPFN-CFM outperforms every baseline in the interventional setting even without the graph supplied, which is where causal knowledge matters. Third, supplying the true graph improves interventional and counterfactual predictions but leaves observational predictions unchanged — the pattern predicted by Theorem 2, since edge orientation does not alter the observational distribution. Performance degrades only mildly out of distribution, indicating the model is not overfitted to its prior.

Table 2 summarizes structure prediction, where TabPFN-CFM outperforms all baselines on adjacency, ancestral and confounding recovery, with the margin largest on the ancestral matrix. As an ablation of the decoder design, we also derive an ancestral matrix from the predicted adjacency matrix by Monte Carlo sampling (Table 10). This is less accurate at every fit-set size — 0.865 versus 0.885 accuracy at $n = 1024$ — because errors in individual edges compound along paths, so predicting long-range ancestry directly is worth the extra decoder head. Full results across fit-set sizes are reported in Tables 11 to 14.

6.3 Real data

Synthetic benchmarks can only demonstrate consistency with the prior. To test transfer, we evaluate on two real datasets with established causal graphs and DoWhy-generated counterfactuals, taken

Table 3: Outcome prediction on real datasets, by mean squared error (↓). Baselines can use neither the true causal graph nor counterfactual information.

	Amazon Sales			Law School Admissions		
Model	Obs.	Int.	Count.	Obs.	Int.	Count.
S-learner	0.002	7.269	-	0.871	0.931	-
T-/X-learner	**0.001**	0.964	-	0.870	0.928	-
DR-learner	0.011	8.716	-	0.872	0.934	-
Do-PFN	0.006	0.593	-	0.893	0.900	-
CausalPFN	0.002	0.967	-	**0.868**	0.925	-
TabPFN-CFM	0.016	**0.515**	0.546	0.871	0.892	**0.120**
TabPFN-CFM w/ graph	0.016	0.523	**0.529**	0.871	**0.885**	**0.120**

Table 4: Effect of the structure loss on outcome prediction. Both models share the architecture, prior, data and a 300k-step training budget, differing only in λ. Prediction loss is the validation cross entropy (↓); query columns report mean squared error (↓), averaged over fit-set sizes.

Training objective	Pred. loss	Obs.	Int.	Count.
Prediction only ($\lambda = 0$)	-0.594	0.463	0.646	0.306
Prediction + structure ($\lambda > 0$)	$\mathbf{-0.608}$	**0.461**	**0.629**	**0.291**

from Robertson et al. (2025): the Amazon Sales dataset (Blöbaum et al., 2024) and the Law School Admissions dataset (Wightman, 1998). The model is applied zero-shot; it has never seen real data during training.

Outcome results are given in Table 3. The pattern from the synthetic experiments repeats: all models are close on the observational query, while TabPFN-CFM leads clearly on the interventional query, reducing MSE from 0.593 to 0.515 on Amazon Sales relative to the strongest baseline. Some meta-learners degrade badly under intervention, the S-learner and DR-learner reach MSEs of 7.27 and 8.72 on Amazon Sales despite near-perfect observational fits illustrating the failure mode that motivates causal modeling: a good associative fit is no guarantee of interventional validity. On Law School Admissions the counterfactual query improves MSE from 0.892 interventionally to 0.120, because the observed factual outcome pins down most of the unit-level exogenous noise. No baseline can exploit this information at all.

Structure results are given in Table 8 in Section H. No baseline is reliable across both datasets: AVICI recovers ancestry well on Amazon Sales, which has a shallow graph well matched to its prior, but is much worse on Law School Admissions, while Arrow is the strongest method on Law School Admissions and no better than chance on Amazon Sales. TabPFN-CFM is the only method ranked first or second in every column. Several classical methods are close to chance, reflecting how weakly their assumptions hold on real data.

6.4 Ablations

Does the structure loss help outcome prediction? Our central claim is that structure and outcome are two readouts of the same latent SCM, so supervising both should improve each. We test this by retraining with $\lambda = 0$ in $\mathcal{L} = \mathcal{L}_{\text{pred}} + \lambda\mathcal{L}_{\text{graph}}$, which removes the structure gradient while leaving the architecture, prior, data and training budget unchanged; the structure decoder is kept but unsupervised, so the comparison isolates the training signal rather than the extra parameters. Table 4 shows the jointly trained model is better on every outcome metric, and that the margin grows with the causal depth of the query: observational MSE is essentially unchanged (0.463 to 0.461), whereas interventional MSE improves by 2.7% and counterfactual MSE by 4.9%. This is the ordering the causal hierarchy predicts. The observational query is associative and does not depend on edge orientation, so structural supervision has nothing to contribute; the interventional and counterfactual queries require knowing which variables are causes of the outcome and which are confounded with it, which is what the structure loss supplies. The structure decoder therefore acts as a regularizer on the

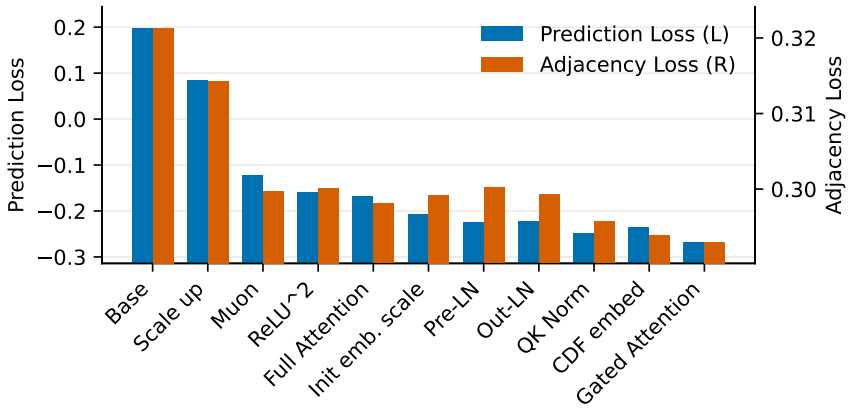

Figure 2: Cumulative ablation over 30k training steps, for prediction loss and adjacency loss. Changes are applied sequentially from left to right, in the order listed in Section 5.4.

shared representation rather than as a competing objective, and structure predictions come at no cost to outcome accuracy.

Backbone and optimizer changes. We next isolate the effect of the backbone and optimizer changes of Section 5.4, which are independent of the causal contributions and add no new functionality. Applying each change cumulatively over 30k training steps, with data held fixed across runs, both the prediction loss and the adjacency loss decrease monotonically (Figure 2); these changes interact nonlinearly and some only show up over longer runs, so the per-change attribution should be read as indicative. We then compare full-length runs with and without the changes, scaling the baseline up to the same 23.2M parameters for fairness, over 150k steps (Figure 7). Our optimized model reaches the same prediction loss $4\times$ faster (37k versus 150k steps to reach -0.195) and the same adjacency loss $3\times$ faster (50k versus 150k steps to reach 0.283), with a lower final loss in both cases. Cost per step rises by roughly 20% (8.52 to 10.24 hours for the full run), largely from Muon and the additional norm layers, so the net saving in compute to reach a target loss is close to $3\times$.

7 Discussion and Conclusion

We have introduced TabPFN-CFM, a causal foundation model that unifies causal structure learning and causal outcome estimation in a single amortized network. Trained purely on a synthetic prior over structural causal models, it recovers acyclic directed mixed graphs, including the bidirected edges that signal unobserved confounding, and answers observational, interventional and counterfactual queries, optionally conditioned on a known graph. Across in-distribution, out-of-distribution and real benchmarks it matches strong tabular baselines associatively while outperforming them interventionally, and outperforms both classical and amortized structure learners on graph recovery. We demonstrate that causal knowledge improves performance, both in training and at inference when graph conditioning helps on interventional and counterfactual queries.

Outlook. A strength of the amortized formulation is that its reach is set by the prior rather than by the architecture, so capability grows with the generative process rather than requiring a new model. Continuous and multi-valued treatments, interventional input data, and cyclic or temporal structures are all natural extensions that can be considered. The training efficiency gains described should also be applicable to new setups and aid the development of new PFN-based models.

AI use statement

We used generative AI tools as a general-purpose writing this paper and as a coding assistant. Generative AI was not used to generate research ideas, to design the method, or to produce experimental

results. The model code is primarily written by the authors. All AI-assisted text and code was reviewed, verified and tested by the authors.

Reproducibility statement

The synthetic prior is fully specified in Section 4 and Section D, including the sampled distributions for graph structure, nonlinearities (Table 5) and noise (Table 6). The architecture is given in Section 5 with full equations and tensor shapes in Section E, and the training objective, graph-dropout scheme and optimizer settings in Sections 5.3 and 5.4. Proofs of both propositions are given in Sections B and C. Evaluation uses 500 cached datasets per setting so that all methods are scored on identical data, and the OOD generator is specified in Section G. Code for data generation, training and evaluation is included in the supplementary materials.

Ethics statement

This work uses synthetic data and two publicly available datasets with established causal graphs. The Law School Admissions dataset contains sensitive attributes including race, and we use it solely as a benchmark for counterfactual prediction, we make no claims about admissions policy. More broadly, causal predictions are often used to make real world decisions. Our work can help improve the accuracy of causal predictions and reduce unwanted or harmful consequence of incorrect assumptions. However, we caution that model predictions may still be incorrect, and should be taken cautiously.

References

Vahid Balazadeh, Hamidreza Kamkari, Valentin Thomas, Benson Li, Junwei Ma, Jesse C. Cresswell, and Rahul G. Krishnan. Causalpfn: Amortized causal effect estimation via in-context learning. In *Advances in Neural Information Processing Systems*, 2025.

Keith Battocchi, Eleanor Dillon, Maggie Hei, Greg Lewis, Paul Oka, Miruna Oprescu, and Vasilis Syrgkanis. Econml a python package for ml-based heterogeneous treatment effects estimation, 2019. URL `https://github.com/py-why/EconML`.

Patrick Blöbaum, Peter Götz, Kailash Budhathoki, Atalanti A. Mastakouri, and Dominik Janzing. Dowhy-gcm: An extension of dowhy for causal inference in graphical causal models, 2024.

D.M. Chickering. Optimal structure identification with greedy search. journal of machine learning research, 2002.

Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *International Conference on Artificial Intelligence*, 2010.

Alex Henry, Prudhvi Raj Dachapally, Shubham Shantaram Pawar, and Yuxuan Chen. Query-key normalization for transformers. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, 2020.

Noah Hollmann, Samuel Müller, Katharina Eggensperger, and Frank Hutter. TabPFN: A transformer that solves small tabular classification problems in a second. In *International Conference on Artificial Intelligence*, 2023.

Guido W. Imbens and Donald B. Rubin. *Causal Inference for Statistics, Social, and Biomedical Sciences: An Introduction*. Cambridge University Press, 2015.

Keller Jordan, Jeremy Bernstein, Brendan Rappazzo, @fernbear.bsky.social, Boza Vlado, You Jiacheng, Franz Cesista, Braden Koszarsky, and @Grad62304977. modded-nanogpt: Speedrunning the nanogpt baseline, 2024a.

Keller Jordan, Yuchen Jin, Vlado Boza, You Jiacheng, Franz Cesista, Laker Newhouse, and Jeremy Bernstein. Muon: An optimizer for hidden layers in neural networks, 2024b.

Nan Rosemary Ke, Silvia Chiappa, Jane X Wang, Jorg Bornschein, Anirudh Goyal, Melanie Rey, Theophane Weber, Matthew Botvinick, Michael Curtis Mozer, and Danilo Jimenez Rezende. Learning to induce causal structure. In *International Conference on Learning Representations*, 2023.

Sören R. Künzel, Jasjeet S. Sekhon, Peter J. Bickel, and Bin Yu. Metalearners for estimating heterogeneous treatment effects using machine learning, 2019.

Jingyuan Liu, Jianlin Su, Xingcheng Yao, Zhejun Jiang, Guokun Lai, Yulun Du, Yidao Qin, Weixin Xu, Enzhe Lu, Junjie Yan, Yanru Chen, Huabin Zheng, Yibo Liu, Shaowei Liu, Bohong Yin, Weiran He, Han Zhu, Yuzhi Wang, Jianzhou Wang, Mengnan Dong, Zheng Zhang, Yongsheng Kang, Hao Zhang, Xinran Xu, Yutao Zhang, Yuxin Wu, Xinyu Zhou, and Zhilin Yang. Muon is scalable for llm training, 2025.

Lars Lorch, Scott Sussex, Jonas Rothfuss, Andreas Krause, and Bernhard Schölkopf. Amortized inference for causal structure learning. In *Advances in Neural Information Processing Systems*, 2022.

Judea Pearl. *Causality : models, reasoning, and inference*. Cambridge University Press, 2000.

Arik Reuter, Anish Dhir, Cristiana Diaconu, Jake Robertson, Ole Ossen, Frank Hutter, Adrian Weller, Mark van der Wilk, and Bernhard Schölkopf. Use what you know: Causal foundation models with partial graphs. In *International Conference on Machine Learning*, 2026.

Thomas Richardson. Markov properties for acyclic directed mixed graphs, 2003.

Jake Robertson, Arik Reuter, Siyuan Guo, Noah Hollmann, Frank Hutter, and Bernhard Schölkopf. Do-PFN: In-context learning for causal effect estimation. In *Advances in Neural Information Processing Systems*, 2025.

Shohei Shimizu, Patrik O. Hoyer, Aapo Hyvärinen, and Antti Kerminen. A linear non-gaussian acyclic model for causal discovery, 2006.

P. Spirtes, C. Meek, and T. Richardson. Causal inference in the presence of latent variables and selection bias, 1995.

Peter Spirtes, Clark N. Glymour, Richard Scheines, and David Heckerman. Causation, prediction, and search, 2000.

Ryan Thompson, He Zhao, Daniel M. Steinberg, and Edwin V. Bonilla. Arrow: A foundation model for causal discovery, 2026.

Linda F. Wightman. Lsac national longitudinal bar passage study, 1998.

Ruibin Xiong, Yunchang Yang, Di He, Kai Zheng, Shuxin Zheng, Huishuai Zhang, Yanyan Lan, Liwei Wang, and Tie-Yan Liu. On layer normalization in the transformer architecture, 2020.

Zhengyan Zhang, Yixin Song, Guanghui Yu, Xu Han, Yankai Lin, Chaojun Xiao, Chenyang Song, Zhiyuan Liu, Zeyu Mi, and Maosong Sun. Relu^2 wins: Discovering efficient activation functions for sparse llms, 2024.

Qingyuan Zhao. On statistical and causal models associated with acyclic directed mixed graphs, 2025.

A Causal Graphs and Structural Causal Models

A *nonparametric structural equation model with independent errors* (NPSEM-IE) satisfies the following conditions (Pearl, 2000):

1. For each $i = 1, \ldots, d$, there exists a set of parent variables $\mathrm{Pa}_i \subseteq V \setminus \{V_i\}$ such that
$$V_i = f_i\big(\mathrm{Pa}_i, U_i\big).$$
2. The graph with vertex set V and directed edges $V_j \to V_i$ whenever $V_j \in \mathrm{Pa}_i$ is acyclic.
3. The unobserved variables are mutually independent and exogenous:
$$U_i \perp\!\!\!\perp U_j \quad \text{for } i \neq j.$$

If our causal system satisfies an NPSEM-IE (or can be trivially reduced to one), corresponding to only "noise" variables being unobserved, then the causal graph corresponding to observed variables V will be a DAG structure. Each variable is independent of its non-descendants given its parents (the local Markov property), with the direction of edges indicating the direction of causality.

However, in the more general case of a DAG over variables (U, V), the NPSEM-IE assumptions need not hold. The unobserved variables in U may be endogenous (i.e., have parents in V or in U), a single unobserved variable may have multiple children in V, and the unobserved variables may be statistically dependent. After marginalizing out U, the causal relationships among the observed variables V can be represented by an acyclic directed mixed graph (ADMG) (Richardson, 2003), which contains both directed edges (indicating directed causal influence) and bidirected edges (indicating the presence of unobserved common causes, i.e. unobserved confounding). Such cases are common in real-world scenarios, where not all relevant variables can be observed or measured.

There are many models referred to as ADMGs in the literature (Zhao, 2025); here we will refer to ADMGs as DAGs with unobserved variables marginalized out. An ADMG graph consists of an adjacency matrix, A, and correlation matrix, C. The adjacency matrix contains directed causal relationships, while the symmetric correlation matrix contains bidirected non-causal correlations caused by unobserved confounding.

Finally, we note there is a more general case, where the underlying structure is not a DAG and instead contains cycles. Such structures can arise in systems with temporal feedback loops or cyclic causality. However, in this work we focus on the acyclic case (DAGs and ADMGs) and leave cyclic structures for future work.

B Proof of Theorem 2: estimation with and without graph input

Including the graph input can improve model predictions. For simplicity, ignore c here. With and without $\mathcal{G}$, the target distributions are

$$p_0(y|D^{\text{obs}}) = \int p(y|\psi)p(\psi|D^{\text{obs}})\, d\psi \tag{3}$$

$$p_G(y|D^{\text{obs}}, \mathcal{G}) = \int p(y|\psi)p(\psi|D^{\text{obs}}, \mathcal{G})\, d\psi \tag{4}$$

$$\tag{5}$$

Comparing their losses, we have:

$$\begin{aligned} &E_{y|D^{\text{obs}},\mathcal{G}}\left[-\log p_G(y|D^{\text{obs}}, \mathcal{G}) - (-\log p_0(y|D^{\text{obs}}))\right] \\ &= \int -p_G(y|D^{\text{obs}}, \mathcal{G}) \log \frac{p_G(y|D^{\text{obs}}, \mathcal{G})}{p_0(y|D^{\text{obs}})}\, dy \qquad (6) \\ &= -KL(p_G(y|D^{\text{obs}}, \mathcal{G})||p_0(y|D^{\text{obs}})) \leq 0 \qquad (7) \end{aligned}$$

Now, taking expectation over D^{obs} and $\mathcal{G}$, we have

$$\begin{aligned} L_G - L_0 &= E_{y,D^{\text{obs}},\mathcal{G}}\left[-\log p_G(y|D^{\text{obs}}, \mathcal{G}) - (-\log p_0(y|D^{\text{obs}}))\right] \qquad (8) \\ &= -E_{y,D^{\text{obs}},\mathcal{G}}\left[KL(p_G(y|D^{\text{obs}}, \mathcal{G})||p_0(y|D^{\text{obs}}))\right] \leq 0 \qquad (9) \end{aligned}$$

Therefore, including the graph input can only improve the model predictions, with no improvement occurring if $p_G(y|D^{\text{obs}}, \mathcal{G}) = p_0(y|D^{\text{obs}})$. Equivalently, in terms of the posterior of ψ, there is an improvement if

$$\int p(y|\psi)p(\psi|D^{\text{obs}}, \mathcal{G})\, d\psi \neq \int p(y|\psi)p(\psi|D^{\text{obs}})\, d\psi \tag{10}$$

$$\int p(y|\psi)\left[p(\psi|D^{\text{obs}}, \mathcal{G}) - p(\psi|D^{\text{obs}})\right]\, d\psi \neq 0 \tag{11}$$

$$p(\psi|D^{\text{obs}}, \mathcal{G}) \neq p(\psi|D^{\text{obs}}) \tag{12}$$

$$\tag{13}$$

So an improvement occurs if including $\mathcal{G}$ changes the posterior distribution of ψ (where y has support). The notable case where this does not occur is when the graph structure is uniquely identified by the observational data, $p(\psi|D^{\text{obs}})$ is a delta distribution on the true SCM, and $p(\psi|D^{\text{obs}}, \mathcal{G})$ is the same delta distribution, the SEM is identifiable. In the finite data regime, even if the SEM is identifiable, the posterior may not be perfect, so including the graph input can still improve the model predictions.

C Proof of Theorem 1: equivalence between log probability and KL divergence

We prove that minimizing log probability (and cross-entropy) equals minimizing KL divergence. This proof extends the proof given in TabPFN (Hollmann et al., 2023) and Do-PFN (Robertson et al., 2025) by including the conditioning on the graph input. We have the following definitions and properties:

$$\psi \sim p(\psi) \tag{14}$$

$$D^{\text{obs}} \sim p(D^{\text{obs}}|\psi) \tag{15}$$

$$\mathcal{G} \sim p(\mathcal{G}|\psi) \tag{16}$$

$$c \sim p(c|\psi) \tag{17}$$

$$y \sim p(y|c, \psi) \tag{18}$$

$$L = -E[\log \hat{p}_\theta(y|c, D^{\text{obs}}, \mathcal{G})] \tag{19}$$

$$\{c, y\} \perp\!\!\!\perp D^{\text{obs}}|\psi \tag{20}$$

$$\{c, y\} \perp\!\!\!\perp \mathcal{G}|\psi \tag{21}$$

$$\tag{22}$$

Optimizing the loss function L is equivalent to optimizing the KL divergence between the true distribution and the model distribution. We have,

$$L = -E[\log \hat{p}_\theta(y|c, D^{\text{obs}}, \mathcal{G})] \tag{23}$$

$$= -\int\int\int \log \hat{p}_\theta(y|c, D^{\text{obs}}, \mathcal{G})p(y, c, D^{\text{obs}}, \mathcal{G})\, dy\, dc\, dD^{\text{obs}} d\mathcal{G} \tag{24}$$

$$= -\int\int\int \log \hat{p}_\theta(y|c, D^{\text{obs}}, \mathcal{G})p(y|c, D^{\text{obs}}, \mathcal{G})p(c, D^{\text{obs}}, \mathcal{G})\, dy\, dc\, dD^{\text{obs}} d\mathcal{G} \tag{25}$$

$$= -E_{c, D^{\text{obs}}, \mathcal{G}}\left[\int p(y|c, D^{\text{obs}}, \mathcal{G}) \log \hat{p}_\theta(y|c, D^{\text{obs}}, \mathcal{G})\, dy\right] \tag{26}$$

$$= -E_{c, D^{\text{obs}}, \mathcal{G}}\left[\int p(y|c, D^{\text{obs}}, \mathcal{G})\left[\log \frac{\hat{p}_\theta(y|c, D^{\text{obs}}, \mathcal{G})}{p(y|c, D^{\text{obs}}, \mathcal{G})} + \log p(y|c, D^{\text{obs}}, \mathcal{G})\right]\, dy\right] \tag{27}$$

$$= E_{c, D^{\text{obs}}, \mathcal{G}}\left[KL(p(y|c, D^{\text{obs}}, \mathcal{G})||\hat{p}_\theta(y|c, D^{\text{obs}}, \mathcal{G}))\right] + C \tag{28}$$

hence minimizing the log probability is equivalent to minimizing the KL divergence between the true distribution and the model distribution. This can also be written with ψ more explicitly. Using the

independence property, we have

$$p(y|c, D^{\text{obs}}, \mathcal{G}) = \int p(y|c, D^{\text{obs}}, \mathcal{G}, \psi) p(\psi|c, D^{\text{obs}}, \mathcal{G})\, d\psi \tag{29}$$

$$= \int p(y|c, \psi) p(\psi|c, D^{\text{obs}}, \mathcal{G})\, d\psi. \tag{30}$$

$$\tag{31}$$

Hence,

$$L = -E_{c, D^{\text{obs}}, \mathcal{G}} \left[\int p(y|c, D^{\text{obs}}, \mathcal{G}) \log \hat{p}_\theta(y|c, D^{\text{obs}}, \mathcal{G})\, dy \right] \tag{32}$$

$$= -E_{c, D^{\text{obs}}, \mathcal{G}} \left[\int \int p(y|c, \psi) p(\psi|c, D^{\text{obs}}, \mathcal{G}) \log \hat{p}_\theta(y|c, D^{\text{obs}}, \mathcal{G})\, dy\, d\psi \right] \tag{33}$$

$$= -E_{c, D^{\text{obs}}, \mathcal{G}, \psi} \left[\int p(y|c, \psi) \log \hat{p}_\theta(y|c, D^{\text{obs}}, \mathcal{G})\, dy \right] \tag{34}$$

$$= E_{c, D^{\text{obs}}, \mathcal{G}, \psi} \left[KL(p(y|c, \psi) || \hat{p}_\theta(y|c, D^{\text{obs}}, \mathcal{G})) \right] + C \tag{35}$$

D Dataset generation

D.1 Generating random SCMs

Firstly, the DAG structure is generated. The DAG size, average connectivity degree and type of DAG are sampled. The type of DAG is either Erdos-Renyi or Scale-Free. For Scale-Free graphs, the attractiveness parameter is also sampled. Given these parameters, a random DAG is generated. A treatment node and target node are sampled, with bias towards selecting the treatment as a causal ancestor of the target to ensure meaningful interventions.

The structural equations are sampled for every node in the DAG. First, the nonlinearity is sampled from a set of possible functions, including whether to use an MLP (Table 5). Then, the type of noise, position of noise and noise distribution are sampled (Table 6). Relevant distribution parameters (e.g. variance) are also sampled randomly. Finally, the node's weight vector $\mathbf{W}$ and bias scalar b are drawn from the Xavier Normal distribution (Glorot & Bengio, 2010).

The node's structural equation takes as input the concatenation of its parent values, $\mathbf{x}_{\text{pa}}$, and computes the node's value in the most general form as:

$$x_{\text{out}} = g(\mathbf{W}_2 f_{\text{nonlinear}}(\mathbf{W}_1 \cdot g(\mathbf{x}_{\text{pa}}, \epsilon_{\text{pre}}) + b), \epsilon_{\text{post}}) \tag{36}$$

Where f is the elementwise nonlinearity, g is the noise composition function (either elementwise additive or multiplicative), ϵ_{pre} and ϵ_{post} are noise variables sampled from the pre and post noise distribution. If the node has no parents, $\mathbf{x}_{\text{pa}}$ is empty and only noise is used as input to the node.

The treatment node is generated differently, since it is a binary variable. The treatment node's structural equation is:

$$\tilde{x}_{\text{out}} = \mathbf{W} \cdot (\mathbf{x}_{\text{pa}} + \epsilon_{\text{pre}}) \tag{37}$$

$$x_{\text{out}} = a \cdot \mathbb{1}[\tilde{x}_{\text{out}} > x_{\text{thresh}}] + b \tag{38}$$

where a, b are random scalars and x_{thresh} is set so the observational distribution of $\tilde{x}_{\text{out}}$ has a randomly sampled class balance. This ensures the treatment always contains positive and negative samples.

Finally, a subset of nodes in the DAG are selected as observed variables V, with the remaining nodes treated as unobserved variables U. The selection is random, with the constraint that the treatment and target nodes are always observed. The probability of a node being unobserved is sampled per SCM, up to a maximum of 0.5 for the training prior. Unobserved variables are marginalized out when generating datasets, creating an observed acyclic adjacency matrix, A, and bidirected confounding matrix, C, between observed nodes together forming the ADMG structure. If the sample is trivial (e.g. all treatments the same or no causal relations), the sample is discarded. This completes the specification of a single SCM, ψ. Our synthetic prior distribution of SCMs, $P(\psi)$, is the distribution induced by the above generative process with all parameters sampled randomly.

D.2 SAMPLING CAUSAL DATASETS FROM SCMS

Now, we describe how each sample dataset is generated. A SCM ψ is sampled from the prior $P(\psi)$. Then, for each sample i in the dataset, the exogenous noise variables U are sampled from their respective distributions. The observed variables V are generated by evaluating the structural equations in topological order. This process is repeated n times to generate the observational dataset with n entries, $\mathcal{D}^{\text{fit}} = \{\mathbf{x}_i, t_i, y_i\}_{i=1}^{n}$. The underlying graph is stored in $G^{\text{est}} = \{A, C\}$.

Then, we generate the prediction dataset. Another sample is drawn from the observational dataset as above, $\mathcal{D}^{\text{pred}} = \{\mathbf{x}^*, t^*, y^*\}$. Since we know the true SCM, the counterfactual sample can also be generated. However, we must be careful to correctly simulate the counterfactual since nodes between T and Y (observed or unobserved) may be descendants of T; these nodes should not be fixed. To account for this, we fix every node and noise variable that may be topologically ordered before T. That is, we split the nodes into descendants of T, N_{desc}, and non-descendants, N_{nd}. All of the noise variables and nodes in N_{nd} are kept fixed to their sampled values from the observational sample. The treatment node is set to its interventional value $T = \text{do}(1 - t^*)$. Nodes in N_{desc} are recomputed with interventional T. This procedure generates the "what-if" counterfactual outcome. Nodes in V between T and Y will change under intervention, but their counterfactual values will depend on their observed value through fixed parent noise variables, allowing the model to use this information to make more accurate predictions. We denote this new sample as $\mathcal{D}^{\text{causal}} = \{\mathbf{x}_t^*, t = \text{do}(1 - t^*), y_t^*\}$. We note there are alternative methods of generating causal queries, such as fixing only ancestors of T corresponding to interventional rather than counterfactual distribution. These are not used in this paper, but we allow users to easily generate and train new models on these alternative distributions with our codebase by changing a configuration flag.

Our model is trained with up to $n = 1600$ fit samples and $k = 33$ covariates (35 nodes in total).

Table 5: Nonliearities. Outputs are scaled to be approxmately standardized. Quantize and Square nonlinearities modified for stability.

Name	Function
Identity	x
Relu	$1.67 \cdot \text{relu}(0, x) - 0.688$
Tanh	$1.43 \cdot \tanh(x)$
Sine	$1.43 \cdot \sin(\pi x)$
Quantize (stabilized)	$\lceil x + \epsilon \cdot \text{rand}(0, 1) \rceil$
Square (stabilized)	$5.7 * \log(1 + 1/5.7 * x^2) - 0.83$

Table 6: Noise generation parameters

Distribution	Position	Composition
Normal	Pre-nonlinearity	Multiplicative
Uniform	Post-nonlinearity	Additive
Bernoulli	Both	
Laplace		
Gamma		

E MODEL ARCHITECTURE DETAILS

This appendix gives the full equations and tensor shapes for the architecture summarized in Section 5. Our architecture is based on TabPFNv2 and Do-PFN, with several upgrades to handle the additional objectives and improve training efficiency. The model consists of four components: the encoder module, the observation module, the graph module and the prediction module.

E.1 ENCODER MODULE

The encoder module embeds the inputs. For notational simplicity, we define / describe the shapes as follows, $\mathbf{x}^{\text{fit}} \in \mathbb{R}^{n\times k}$, $\mathbf{x}^* \in \mathbb{R}^{1\times k}$, $t^{\text{fit}} \in \{0,1\}^n$, $t^* \in \{0,1\} \times \{0,1\}$, $y^{\text{fit}} \in \mathbb{R}^n$, $G \in \{0,1\}^{(k+2)\times(k+2)}$, where k is the number of covariates, and n is the number of observations. The hidden dimension of the transformer is h. Both $\mathbf{x}^{\text{fit}}$ and $\mathbf{x}^*$ are encoded with the same linear transform, followed by a column-wise positional embedding:

$$h_a^x = \text{Linear}_x(\mathbf{x}||\text{CDF}(\mathbf{x})) \in \mathbb{R}^{n\times k\times h} \tag{39}$$

$$h^x = h_a^x + \text{PosEmb}_{\text{col}} \in \mathbb{R}^{n\times k\times h}. \tag{40}$$

Each covariate is concatenated with its CDF transform, which helps embedding extreme outliers. Each of the k feature columns of the dataset is given a positional embedding generated by the subspace method from TabPFNv2. For each column j

$$\text{PosEmb}_{\text{col } j} = W_{\text{col}}\epsilon \tag{41}$$

$$\epsilon \sim \mathcal{N}(0, I_{h/4}) \tag{42}$$

$$W_{\text{col}} \in \mathbb{R}^{h\times h/4}. \tag{43}$$

One embedding is generated for each of the k columns and concatenated together. A new positional embedding is generated for each forward pass, meaning the positional embeddings are randomized, for both training and inference. The targets y^{fit} and y^* are embedded with a linear transform,

$$h^y = \text{Linear}_y(y||\text{CDF}(y)||I_{\text{type}}) \in \mathbb{R}^{n\times h} \qquad y \in \{y^{\text{fit}}, y^*\} \tag{44}$$

where $I_{\text{type}} \in \{0,1,2\}$ is an indicator for the type of y, with value 0 for y^{fit}, 1 for counterfactual queries where y^* is observed, and 2 for counterfactual queries where y^* is unobserved. The CDF embedding is used here again. Next, we embed the treatment variables,

$$h^t = \text{emb}_t(t^{\text{fit}}) \in \mathbb{R}^{n\times 1\times h} \tag{45}$$

$$h^{t*} = \text{emb}_{t*}(t^*) \in \mathbb{R}^{1\times 1\times h}. \tag{46}$$

Note the embedding for t^* has 4 values corresponding to if the query is interventional or observational. These are concatenated to generate the full input embeddings,

$$h_0^{\text{fit}} = \text{Concat}(h^t, h^x, h^y, \text{dim} = 1) \in \mathbb{R}^{n\times(k+2)\times h} \tag{47}$$

$$h_0^{\text{pred}} = \text{Concat}(h^{t*}, h^{x*}, \mathbf{0}, \text{dim} = 1) \in \mathbb{R}^{1\times(k+2)\times h}. \tag{48}$$

Finally, the prior ADMG graph structure is embedded. The ADMG is represented as a directed adjacency matrix, A, and bidirected correlation matrix C. Both matrices are binary. If the matrices are unknown, they are filled with zeros. From A, we create the ancestral matrix R, the matrix consisting of all ancestors for each node (reachability or transitive closure). The graph embedding is

$$h^G = \text{emb}_A(A) + \text{emb}_R(R) + \text{emb}_C(C) + \text{emb}_{AT}(A^T) + \text{emb}_{RT}(R^T) + \text{emb}_{CT}(C^T). \tag{49}$$

where emb() is an elementwise learned binary embedding. This representation is heavily redundant since the transpose matrices and ancestral matrices are constructed from A and C. This approach allows the model to easily identify every node's parents and children as well as ancestors and descendants without multiple hops.

Unlike TabPFNv2 (and Do-PFN), we omit input augmentations: grouping features, ensembling, random augmentations and random feature products. As well as simplifying the model, this is also required since the structure is explicitly represented in the input, so mixing features would not easily work. However, the CDF augmentation is kept in our model, as a fixed part of the encoder.

E.2 OBSERVATION MODULE

The observation module processes h_0^{fit}. The embeddings are passed through L tabular transformer layers with self attention. Each transformer block consists of a row-wise attention, column-wise

attention and a feed forward layer.

$$h_{\text{row}} = \text{MultiHeadAttention}(h, dim = 1) \tag{50}$$

$$h_{\text{col}} = \text{MultiHeadAttention}(h, dim = 0) \tag{51}$$

$$h_{\text{mlp}} = \text{FFN}(h) \tag{52}$$

$$h_l^{\text{fit}} = \text{Block}(h_{l-1}^{\text{fit}}) \tag{53}$$

where Block denotes a pre-layer-normalized transformer block with residual sublayers values using h_{row}, h_{col}, and h_{mlp} and h is the post layer normalized residual input for each block.

Let $h_{l,i}^{\text{fit}}$ be the output of layer l, feature i. The matrix decoder module uses the final embeddings $h_{L,i}^{\text{fit}}$ to generate predicted graph structure matrices. The embeddings are averaged over fit rows and projected using two linear layers with RMS normalization to map each output to embeddings $u_i, v_i \in \mathbb{R}^h$. The logit for each edge is

$$\hat{M}_{i,j} = \alpha \cdot \text{gelu}(u_i^T \cdot W) \cdot v_j + \beta, \qquad \hat{M} \in \{\hat{A}, \hat{R}, \hat{C}\} \tag{54}$$

with learned parameters $W \in \mathbb{R}^{h \times h}$ and scalar α and β. Separate decoder parameters are used for each output matrix, $\hat{A}, \hat{R}, \hat{C}$. Entries in $\hat{M} \in \mathbb{R}^{(k+2)\times(k+2)}$ are logits representing the probability of each edge occurring. Empirically, we found this nonlinear decoder worked better than a direct dot product used in AVICI (Lorch et al., 2022). The matrix decoder's predicted graph depends only on $\mathcal{D}^{\text{fit}}$, and not on $\mathcal{D}^{\text{pred}}$ or the input graph structure.

E.3 Graph module

The graph module takes the graph embeddings and the intermediate hidden states from the observation module to generate graph hidden states. These hidden states use feature observation hidden states for both positional embeddings and to pass information. The intermediate embeddings h_l^{col} are averaged over the sample dimension to create feature embeddings, $\bar{h}_l^{\text{col}} \in \mathbb{R}^{(k+2)\times h}$. These embeddings are then transformed into source and destination embeddings using two separate linear layers. The resulting representations are combined via an outer product to produce the feature matrix $H_l^{\text{feat}} \in \mathbb{R}^{(k+2)\times(k+2)\times h}$. The graph hidden states are computed as

$$H_l^G = \text{LN}(h^G + H_l^{\text{feat}}). \tag{55}$$

Note that this module only depends on $\mathcal{D}^{\text{fit}}$ and G, not on $\mathbf{x}^*$. Also, h^G is not updated, we always use the initial h^G from the encoder module. Empirical testing found no improvements from updating these weights between each layer.

E.4 Prediction module

The prediction module generates the distribution $\hat{p}_\theta(y)$, using the previous embeddings and encoded prediction inputs. This module follows the same architecture and parameterization as the observation module, with the addition of a cross-attention layer over the graph module embeddings:

$$h_{\text{row}}^* = \text{MultiHeadAttention}(h^*, \text{dim} = 1), \tag{56}$$

$$h_{\text{col}}^* = \text{MultiHeadAttention}(\text{KV} = h_{\text{row}}, \text{Q} = h^*, \text{dim} = 0), \tag{57}$$

$$h_G^* = \text{MultiHeadAttention}(\text{KV} = H^G, \text{Q} = h^*, \text{dim} = 0), \tag{58}$$

$$h_{\text{mlp}} = \text{FFN}(h_G^*), \tag{59}$$

$$h_l^* = \text{Block}(h_{l-1}^*). \tag{60}$$

Block is a pre-layer norm residual block using $h_{\text{row}}^*, h_{\text{col}}^*, h_{\text{mlp}}$ residual sublayers, with residual stream h^*. Cross attention to the observation embeddings h_{row} and H^G allows the prediction module to access the observational data and graph structure. Finally, the predicted outcome is generated from the final embedding corresponding to the outcome feature, $h_{L,y}^* \in \mathbb{R}^h$ using a MLP head, mapping to prediction bucket logits,

$$z = \text{MLP}(h_{L,y}^*) \in \mathbb{R}^b \tag{61}$$

$$\hat{p}_\theta(y) = \text{softmax}(z) \in \mathbb{R}^b \tag{62}$$

where z are output logits, b is the number of buckets in the discretized output distribution and $\hat{p}_\theta(y)$ is the predicted probability distribution over these buckets.

F SYNTHETIC TOY EXAMPLES

F.1 LINEAR INSTRUMENTAL VARIABLE EXAMPLE

Showcase experiment on the Instrumental Variable (IV) problem, with SEM: $Z \to T \to Y, U \to T, U \to Y$. To demonstrate, we examine predictions from a simple random synthetic linear SEM. The SEM equations (after normalizing with the observational sample statistics) are:

$$\begin{aligned}\epsilon_{\{u,z,t,y\}} &\sim U(-0.4, 0.4)\\ U &= 4.303\epsilon_u - 0.031\\ Z &= 4.380\epsilon_z - 0.007\\ T &= \text{binarize}_{0.5}(-0.105Z + 0.104U + \epsilon_t)\\ Y &= -1.818 * T - 0.108 * U + 1.047 * \epsilon_y + 0.906\end{aligned}$$

where the binarize function returns 1 if the input is in the top 50 percent of the overall distribution and 0 otherwise. We draw a single sample from this SEM:

$$\begin{aligned}z &= -0.187\\ u &= 0.421\\ t &= 1\\ y &= -0.568\end{aligned}$$

Now, we identify the exact distribution assuming everything is observed, for observational, interventional, and counterfactual queries. The observational distribution of y is

$$Y = -0.957 + 1.047 * \epsilon_y,$$

the interventional distribution with $do(T = 0)$ is

$$Y = 0.861 + 1.047 * \epsilon_y$$

and the counterfactual distribution, which assumes y is observed, is

$$Y = 1.250$$

since $\epsilon_y = 0.372$ can be determined by the observation of y, which allows for the exact counterfactual value to be computed with no uncertainty.

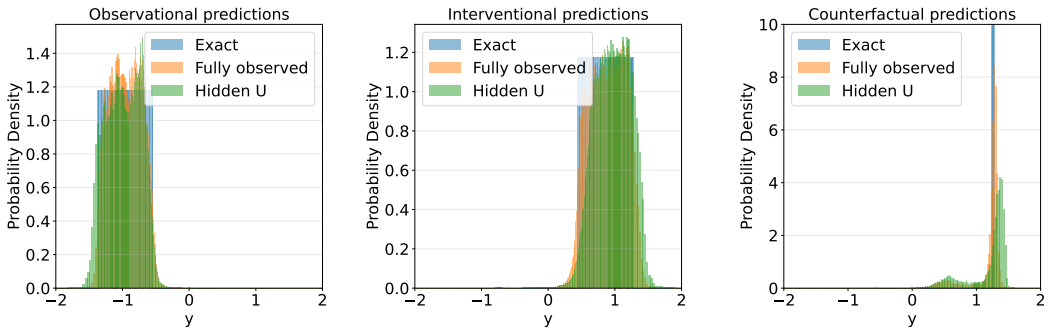

Figure 3: Observational (left), Interventional (center), and Counterfactual (right) distributions for the IV example. Exact solutions are in blue, model predictions in orange for when U is observed, and in green when U is unobserved. Model is *not* given the graph structure.

F.2 NONLINEAR SEM

A more complex nonlinear SEM is used to test the model. The graph structure is fixed as $Z \to T \to V \to Y, U \to V, U \to W \to Y$ where U is unobserved. The graph contains latent variables

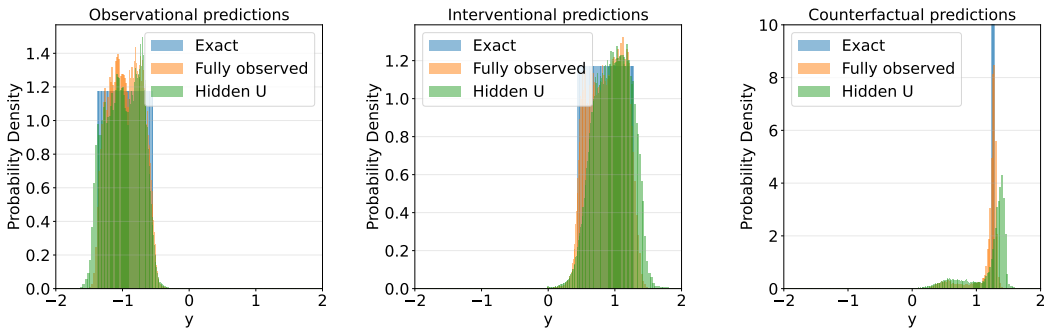

Figure 4: Observational (left), Interventional (center), and Counterfactual (right) distributions. Exact solutions are in blue, model predictions in orange for when U is observed, and in green when U is unobserved. Model is given the graph structure.

and covariates that are descendants of T. The SEM equations are drawn from our nonlinear prior distribution. Since the exact posterior for interventional and observational distributions is intractable, we compare the model predictions to a single sample drawn from the true distribution. Results are shown for $n = 512$. Figure 5 shows the model predictions align well with the samples, even without the graph structure. Giving the graph structure improves predictions. Table 7 shows the true and predicted adjacency matrices. The predicted adjacency matrix generally matches the true adjacency matrix, though there is some uncertainty in the children of T likely due to V, W, Y all being correlated. In Figure 6, we show the predicted distribution shifts closer towards the true counterfactual as n increases (without $\mathcal{G}$). For small n, the counterfactual is biased towards the observed y^*, which vanishes as n increases and the model is able to learn the underlying SEM better.

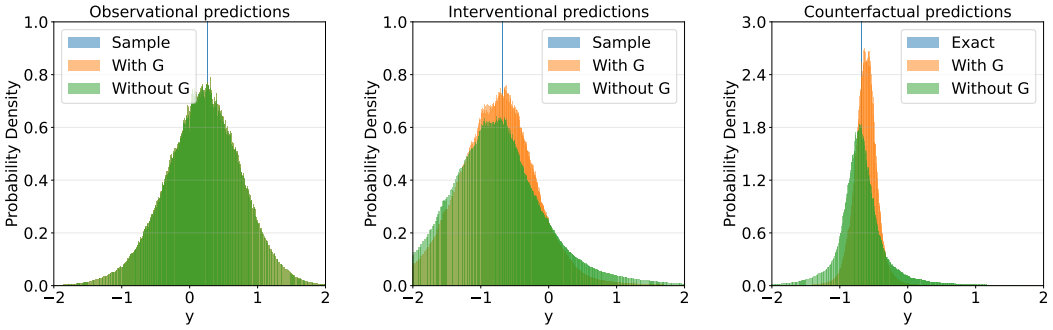

Figure 5: Observational (left), Interventional (center), and Counterfactual (right) distributions. Predictions are compared with and without the true graph structure $\mathcal{G}$. The Observational and Interventional predictions are compared to a single sample drawn from the true distribution, while the Counterfactual predictions are compared to the true counterfactual value.

Table 7: True and predicted adjacency matrices for the nonlinear SEM experiment. Left: True adjacency matrix. Right: Predicted adjacency matrix (values represent predicted edge probabilities).

	T	Z	V	W	Y
T	0	0	1	0	0
Z	1	0	0	0	0
V	0	0	0	0	1
W	0	0	0	0	1
Y	0	0	0	0	0

	T	Z	V	W	Y
T	0.000	0.002	0.992	0.938	0.552
Z	0.982	0.000	0.318	0.522	0.391
V	0.000	0.000	0.000	0.597	0.758
W	0.006	0.005	0.573	0.000	0.871
Y	0.000	0.000	0.001	0.005	0.000

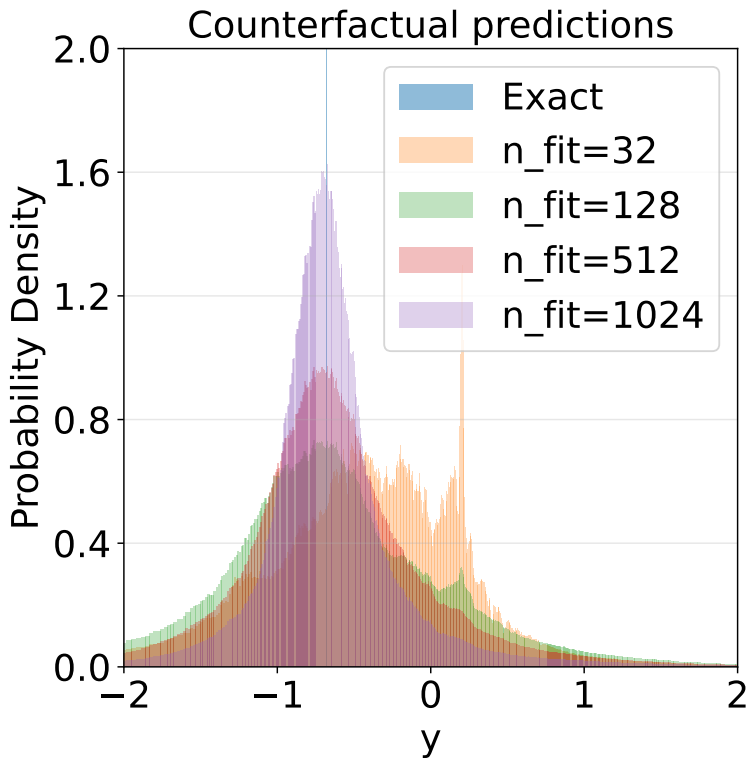

Figure 6: Counterfactual distribution predictions for the nonlinear SEM with different $\mathcal{D}^{\text{fit}}$ sample sizes.

G OOD DATA GENERATION

We test our model on datasets drawn using random Fourier functions as nonlinearities. This has been used as part of the prior in other works, but our models are not trained on these functions, so they are OOD. The distribution of SEM functions is constructed as follows. First, parameters are sampled,

$$w \sim U[0.1, 0.5] \tag{63}$$
$$c \sim U[1.41, 5.65] \tag{64}$$
$$b \sim U[-0.5, 0.5] \tag{65}$$
$$W_{i,j} \sim \text{Cauchy}(0, w) \tag{66}$$
$$\phi_i \sim N(0, 2\pi) \tag{67}$$
$$k_i \sim N(0, c(n_{\text{in}})^{-0.5}). \tag{68}$$

The output of each node is

$$y = b + \sum_{i=1}^{32} k_i \cos\big((xW^\top)_i + \phi_i\big) \tag{69}$$

where x is a concatenation of the node's parent values and noise variables, and parameters W, k, ϕ are randomly sampled frequency, magnitude and phase for each of the 32 coefficients. This process generates random functions that exhibit non-trivial nonlinear behaviors over a range of frequencies.

We additionally shift the graph distribution away from the training prior. The maximum probability that a node is designated unobserved is raised from 0.5 to 0.75, so a larger fraction of nodes is marginalized out. This yields sparser observed graphs with fewer covariates and more bidirected edges, pushing the evaluation further out of distribution along the structural axis as well as the functional one.

H ADDITIONAL RESULTS

H.1 CUMULATIVE ARCHITECTURE ABLATION

Ablation runs use smaller datasets and 30k training steps, with training and validation data held fixed across all setups to isolate the effect of the architecture changes. The learning rate is 3e-4 for Muon and 1e-4 for AdamW, the latter requiring a lower rate for stability. The cumulative ablation is shown in Figure 2.

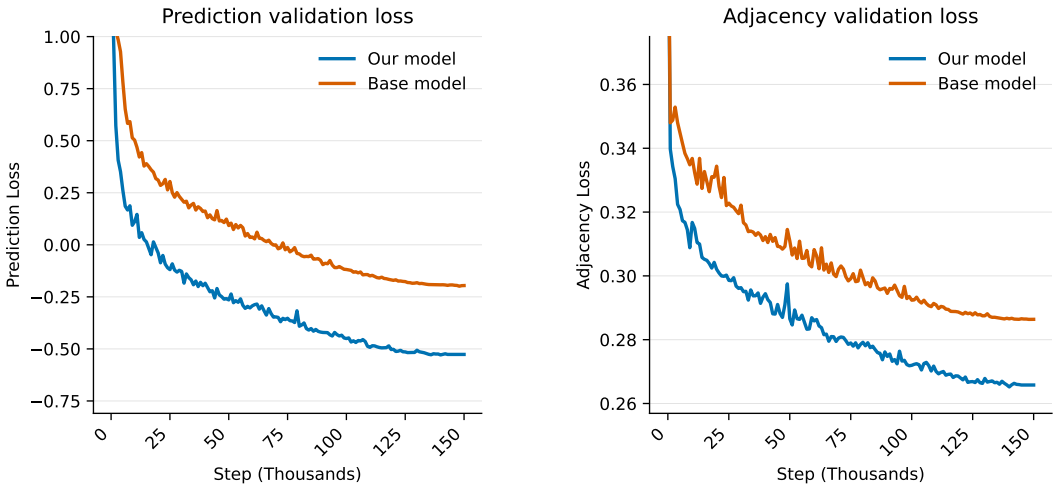

Figure 7: Validation loss over 150k training steps comparing our backbone against the baseline at matched parameter count: prediction loss (left), adjacency loss (right).

Table 8: Structure prediction on real datasets, evaluated by AUROC (↑) with elementwise accuracy in brackets. Neither dataset supplies ground-truth confounding, so only the adjacency and ancestral matrices are scored.

	Amazon Sales		Law School Admissions	
Model	Adjacency	Ancestral	Adjacency	Ancestral
FCI	0.523 (0.79)	0.532 (0.63)	0.500 (0.69)	0.500 (0.63)
GES	0.527 (0.53)	0.603 (0.60)	0.627 (0.56)	0.700 (0.63)
LiNGAM	0.464 (0.59)	0.600 (0.62)	0.353 (0.43)	0.440 (0.48)
PC	0.573 (0.72)	0.555 (0.57)	0.671 (0.56)	0.700 (0.63)
AVICI	0.698 (0.78)	**0.872** (**0.81**)	0.545 (0.69)	0.517 (0.63)
Arrow	0.493 (0.55)	0.366 (0.38)	**0.926** (0.78)	**0.913** (0.71)
TabPFN-CFM	**0.773** (**0.81**)	0.816 (0.78)	0.883 (**0.94**)	0.877 (**0.88**)

H.2 Structure loss ablation

The two runs compared in Table 4 use the final architecture and are trained on identical data for 300k steps, differing only in the weight λ on the structure loss. The ablated run keeps the structure decoder but receives no gradient through it, so the comparison measures the effect of the structural training signal rather than of the additional parameters. Outcome metrics are averaged over fit-set sizes, so the absolute values are not directly comparable with Table 11, which reports each fit-set size separately; only the difference between the two rows should be read.

H.3 Full result tables

Table 9: Predicted adjacency matrices for the linear SEM experiment on the IV graph. Values represent predicted edge probabilities. The true edges are $Z \to T$, $T \to Y$, $U \to T$ and $U \to Y$.

	T	Z	U	Y
T	-	0.078	0.010	0.997
Z	0.873	-	0.208	0.129
U	0.780	0.460	-	0.978
Y	0.000	0.006	0.002	-

Table 10: Comparison of direct ancestral matrix prediction and ancestral matrix generated from the predicted adjacency matrix using Monte-Carlo sampling for out-of-distribution SEMs, across different numbers of fit data points (n-fit). Values represent AUC (↑), with accuracy (↑) in brackets.

n-fit	Direct ancestral	Ancestral from adjacency
32	0.693 (0.82)	0.700 (0.72)
256	0.832 (0.86)	0.826 (0.81)
1024	0.894 (0.89)	0.889 (0.87)

Table 11: Predictions on in distribution SEMs with varying fit samples (second header), compared by mean squared error (↓). The baseline methods are unable to make counterfactual predictions.

	Observational			Interventional			Counterfactual		
Model	32	256	1024	32	256	1024	32	256	1024
S-learner	0.591	0.457	0.430	0.942	0.797	0.797	-	-	-
T-learner	0.645	0.466	0.434	1.209	0.852	0.846	-	-	-
X-learner	0.645	0.466	0.434	1.209	0.852	0.846	-	-	-
DR-Learner	0.645	0.461	0.431	0.987	0.807	0.825	-	-	-
Do-PFN	1.508	0.796	0.670	2.322	1.080	0.941	-	-	-
CausalPFN	0.787	0.483	0.436	1.092	0.822	0.777	-	-	-
TabPFN-CFM	0.578	0.457	0.432	0.781	0.616	0.576	0.410	0.279	0.246
TabPFN-CFM With Graph	0.572	0.456	0.432	0.735	0.580	0.550	0.355	0.239	0.216

Table 12: Structural predictions on in distribution SEMs with varying fit samples. Main values track AUROC (↑), with accuracy (↑) in brackets.

	Adjacency Matrix			Ancestral Matrix		
Model	32	256	1024	32	256	1024
FCI	0.523 (0.85)	0.532 (0.85)	0.533 (0.85)	0.516 (0.56)	0.521 (0.56)	0.536 (0.56)
GES	0.585 (0.82)	0.637 (0.82)	0.672 (0.81)	0.616 (0.67)	0.651 (0.63)	0.657 (0.60)
LiNGAM	0.530 (0.80)	0.556 (0.79)	0.568 (0.78)	0.506 (0.72)	0.524 (0.71)	0.540 (0.70)
PC	0.587 (0.83)	0.632 (0.82)	0.651 (0.81)	0.580 (0.55)	0.637 (0.53)	0.632 (0.53)
AVICI	0.611 (0.83)	0.701 (0.85)	0.703 (0.85)	0.572 (0.77)	0.683 (0.80)	0.682 (0.80)
Arrow	0.688 (0.84)	0.720 (0.85)	0.725 (0.84)	0.678 (0.78)	0.710 (0.79)	0.715 (0.79)
TabPFN-CFM	0.760 (0.87)	0.866 (0.89)	0.898 (0.90)	0.767 (0.82)	0.882 (0.87)	0.916 (0.89)

	Confounding Matrix		
Model	32	256	1024
FCI	0.530 (0.85)	0.554 (0.85)	0.576 (0.84)
GES	-	-	-
LiNGAM	-	-	-
PC	-	-	-
AVICI	-	-	-
Arrow	-	-	-
TabPFN-CFM	0.792 (0.89)	0.840 (0.89)	0.868 (0.89)

Table 13: Predictions on out of distribution SEMs with varying fit samples (second header), compared by mean squared error (↓). The baseline methods are unable to make counterfactual predictions.

	Observational			Interventional			Counterfactual		
Model	32	256	1024	32	256	1024	32	256	1024
S-learner	0.811	0.623	0.565	1.084	0.896	0.888	-	-	-
T-learner	0.864	0.642	0.574	1.283	0.949	0.909	-	-	-
X-learner	0.863	0.462	0.574	1.283	0.949	0.909	-	-	-
DR-Learner	0.850	0.648	0.577	1.104	0.896	0.885	-	-	-
Do-PFN	1.336	0.814	0.782	2.599	1.006	0.966	-	-	-
CausalPFN	1.049	0.707	0.613	1.401	0.955	0.903	-	-	-
TabPFN-CFM	0.748	0.592	0.534	0.969	0.792	0.755	0.813	0.642	0.603
TabPFN-CFM With Graph	0.743	0.591	0.534	0.934	0.768	0.734	0.748	0.607	0.575

Table 14: Structural predictions on synthetic OOD SEMs with varying fit samples. Main values track AUROC (↑), with accuracy (↑) in brackets.

	Adjacency Matrix			Ancestral Matrix		
Model	32	256	1024	32	256	1024
FCI	0.515 (0.84)	0.526 (0.84)	0.535 (0.84)	0.500 (0.59)	0.503 (0.59)	0.504 (0.59)
GES	0.550 (0.83)	0.609 (0.83)	0.654 (0.81)	0.554 (0.73)	0.616 (0.67)	0.654 (0.60)
LiNGAM	0.516 (0.81)	0.544 (0.81)	0.558 (0.78)	0.501 (0.74)	0.520 (0.73)	0.524 (0.70)
PC	0.587 (0.82)	0.656 (0.86)	0.685 (0.84)	0.578 (0.79)	0.628 (0.56)	0.639 (0.56)
AVICI	0.618 (0.83)	0.637 (0.86)	0.623 (0.86)	0.593 (0.77)	0.619 (0.80)	0.596 (0.80)
Arrow	0.701 (0.84)	0.713 (0.85)	0.734 (0.85)	0.686 (0.79)	0.707 (0.80)	0.736 (0.81)
TabPFN-CFM	0.697 (0.86)	0.822 (0.89)	0.880 (0.90)	0.693 (0.82)	0.832 (0.86)	0.894 (0.89)

	Confounding Matrix		
Model	32	256	1024
FCI	0.536 (0.84)	0.572 (0.82)	0.560 (0.81)
GES	-	-	-
LiNGAM	-	-	-
PC	-	-	-
AVICI	-	-	-
Arrow	-	-	-
TabPFN-CFM	0.756 (0.87)	0.789 (0.87)	0.804 (0.86)