Does TabPFN Understand Causal Structures?

Omar Swelam[1] **Lennart Purucker**[1] **Jake Robertson**[3,2,1] **Hanne Raum**[1]
Joschka Boedecker[1,4] **Frank Hutter**[2,3,1]
[1]University of Freiburg [2]Prior Labs [3]ELLIS Institute Tübingen [4] BrainLinks-BrainTools
swelamo@informatik.uni-freiburg.de

Abstract

Causal discovery is fundamental for multiple scientific domains, yet extracting causal information from real world data remains a significant challenge. Given the recent success on real data, we investigate whether TabPFN, a transformer-based tabular foundation model pre-trained on synthetic datasets generated from structural causal models, encodes causal information in its internal representations. We develop an adapter framework using a learnable decoder and causal tokens that extract causal signals from TabPFN's frozen embeddings and decode them into adjacency matrices for causal discovery. Our evaluations demonstrate that TabPFN's embeddings contain causal information, outperforming several traditional causal discovery algorithms, with such causal information being concentrated in mid-range layers. These findings establish a new direction for interpretable and adaptable foundation models and demonstrate the potential for leveraging pre-trained tabular models for causal discovery.

1 Introduction

Traditional causal discovery methods face substantial challenges and impose assumptions that are often unverifiable in practice (Spirtes et al., 2000; Chickering, 2002). Meanwhile, tabular foundation models (TFMs), such as TabPFN (Hollmann et al., 2022, 2025), have shown remarkable performance and generalization on tabular data tasks despite having been pre-trained only on synthetic data. Which raises a question: do the internal representations of TabPFN encode causal knowledge beyond statistical correlations? We investigate whether TabPFN's (specifically TabPFNv2 (Hollmann et al., 2025)) representations contain causal information by developing a framework that learns to extract this information for causal discovery. Our framework employs learnable dual-attention decoder and universal tokens to extract causal signals from TabPFN's data embeddings and decode them into adjacency matrices. Our approach combines insights from tabular prompt-tuning methods (Feuer et al., 2024) as we introduce universal tokens that are tuned to aggregate the causal information via our decoder for a given dataset in-context, in contrast to dataset-specific tuning, while following the neural causal discovery framework of Lorch et al. (2022) in terms of problem formulation.

Contributions. We make three primary contributions: (1) introducing a novel research direction probing tabular foundation models for implicit causal knowledge, (2) developing a causal discovery framework that depends on foundation model representations, and (3) demonstrating that causal information is concentrated in the pre-trained TabPFN's middle layers.

2 Related work and background

Causal discovery Traditional methods exploit observational and interventional data to extract causal structure following different approaches, including constraint-based (Spirtes et al., 2000), score-based (Chickering, 2002), continuous optimization (Zheng et al., 2018), and interventional methods (Hauser

EurIPS 2025 Workshop: AI for Tabular Data.

& Bühlmann, 2012; Wang et al., 2017; Brouillard et al., 2020). While many of these approaches offer strong guarantees, they often make overly strict assumptions about the data generating process, including the nature of the noise signal (Shimizu et al., 2006), and face exponential computational complexity as feature sizes increase, leading to an increasing number of statistical tests (Mokhtarian et al., 2025). In recent years, neural causal discovery methods (Lorch et al., 2022; Ke et al., 2023; Dhir et al., 2025) address these limitations in an end-to-end manner without imposing the same limiting assumptions about data in their design, leveraging transformers trained on synthetic datasets for efficient single-pass causal graph prediction that is scalable to increasing feature sizes. Unlike these methods that are trained specifically for the causal discovery task, we exploit existing foundation models that may encode causal knowledge through their pre-training on predictive tasks.

Prior-Data Fitted Networks and causality Prior-Data Fitted Networks (PFNs) are models pre-trained on synthetic datasets to perform a variety of predictive tasks. TabPFN (Hollmann et al., 2022, 2025) is pre-trained on datasets generated from structural causal models (SCMs) and has achieved state-of-the-art performance for classification and regression. Recent extensions of PFNs to causal inference tasks, including causal fairness and treatment effect prediction (Robertson et al., 2024, 2025; Ma et al., 2025), demonstrate that PFNs can be trained to encode causal representations, which motivates us to investigate whether TabPFNv2, due to its causal prior, encodes causal knowledge.

AVICI Lorch et al. (2022) introduced a causal discovery framework that amortizes causal structure learning. Similar to TabPFN, it employs a dual-attention encoder to process the data, whose embeddings are aggregated into feature-wise representations, each corresponding to a graph node. These representations are used to predict the adjacency matrix in a pairwise manner. We adopt the same approach for adjacency prediction (Section 3.1) and loss function (Section 3.2). To ensure a standardized comparison, we also use the same synthetic data-generating pipeline (Section 3.3).

3 Methodology

3.1 Architecture design

Our proposed architecture is illustrated in Figure 1. We use TabPFNv2's classification backbone encoder with frozen weights and introduce t learnable universal causal tokens $Q_0 \in \mathbb{R}^{t \times f \times d}$, which are prompt-tuned for causal discovery within a learnable dual-attention decoder that mirrors the encoder architecture but differs in its attention source. Our architecture proceeds as follows for a dataset with f features and n samples: (1) We obtain cell-wise d-dimensional representations via TabPFN's frozen embedding layer to produce data embeddings $H_0 \in \mathbb{R}^{n \times f \times d}$. These embeddings pass through the first four layers ($L = 4$) of TabPFN's dual-attention encoder, yielding data tokens H_L. (2) In our learnable decoder, causal tokens attend to these data tokens at each layer, summarizing causal information and producing output tokens $R_L \in \mathbb{R}^{t \times f \times d}$. (3) We aggregate the decoder outputs across the t dimension into k tokens ($k < t$), which are concatenated along the representational dimension to form expressive feature-wise representations $\in \mathbb{R}^{f \times k}$. (4) These representations are linearly projected via learnable matrices $U, V \in \mathbb{R}^{k \times k}$ into parent and child embeddings, and a dot product is applied to predict adjacency entries for each parent–child pair. (5) Finally, we apply a sigmoid activation to obtain predicted edge probabilities. Further details are provided in Appendix A.

3.2 Objective function

The objective is to amortize causal structure inference by maximizing the log-likelihood of the ground-truth adjacency matrix, approximated via binary cross-entropy over the edges represented as binary entries in the matrix. Training also promotes acyclicity of the predicted adjacency matrix, estimated by its spectral radius following Lee et al. (2019), through constrained optimization. Further details are provided in Appendix B.

3.3 Data generation

Synthetic data are generated by sampling a directed acyclic graph (DAG) and then drawing samples from it under causal sufficiency. For each dataset, all parent–child mechanisms are defined using either a linear or a random Fourier feature (RFF) function. Each variable is sampled conditionally on its parents according to the chosen functional type. For details about graph structures, mechanisms, and noise types, see Appendix C.

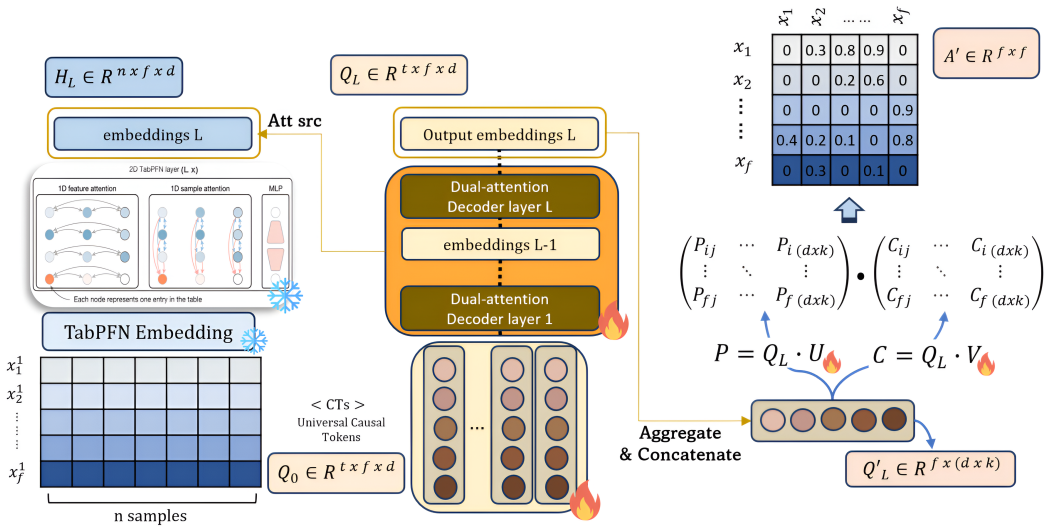

Figure 1: Overall architecture of our approach, where the data embeddings from the frozen TabPFN (left) are attended to in the decoder (middle) to extract aggregated feature-representations for the adjacency matrix prediction (right)

4 Experiments

4.1 Training and evaluation details

The model was trained for 100,000 optimization steps with a batch size of 32 datasets and their corresponding DAGs, using the AdamW optimizer (Loshchilov & Hutter, 2019) with cosine annealing (Loshchilov & Hutter, 2017) (initial learning rate = 5e-4). For each training batch, the feature dimensionality of datasets was randomly sampled between 4 and 20, proportional to the total number of features. Each dataset was generated according to one of two sampling schemes: (i) with probability 0.75, consisting of 100 observational and 100 interventional samples; or (ii) with probability 0.25, consisting solely of 200 observational samples. The number of causal tokens used is $t = 30$, aggregated into $k = 4$ tokens. Our model has 6M parameters, 3.6M of which are learnable.

We evaluated on 500 datasets with their corresponding DAGs, spanning feature sizes of 5, 7, 10, 15, and 20. Each dataset contains 300 observational and 300 interventional samples. Our performance is compared against the pre-trained AVICI (scm-v0) model and statistical baselines for causal discovery, namely GIES (Hauser & Bühlmann, 2012), IGSP (Wang et al., 2017), and DCDI (Brouillard et al., 2020). Due to computational constraints, DCDI was evaluated on only 50 datasets. We report the ROC AUC and AP scores given the probabilistic binary classification nature of our edge predictions.

4.2 Results

Our experiments are geared to answer the following research questions, with the key message that our framework can extract causal information from TabPFN and outperforms statistical baselines in the downstream task of causal discovery.

RQ1: Can we extract causal information from TabPFN embeddings? First, we empirically show that our approach extracts causal information by achieving ROC AUC scores close to AVICI (Figure 2). While we outperform statistical methods on ROC AUC and AP scores, we observe that the AP scores decline at increasing feature sizes for our approach and the statistical baselines, suggesting that while TabPFN encodes causal structure, it struggles to distinguish the correct causal relations as the number of possible edges increases. Appendix D further analyzes how performance varies across graph structures, likely due to TabPFN's pre-training on small and sparse graphs for predictive tasks.

RQ2: How does causal information propagate through TabPFN? Figure 3 shows that relying on data embeddings from the middle layers of TabPFN, namely layers 4-6, results in better causal

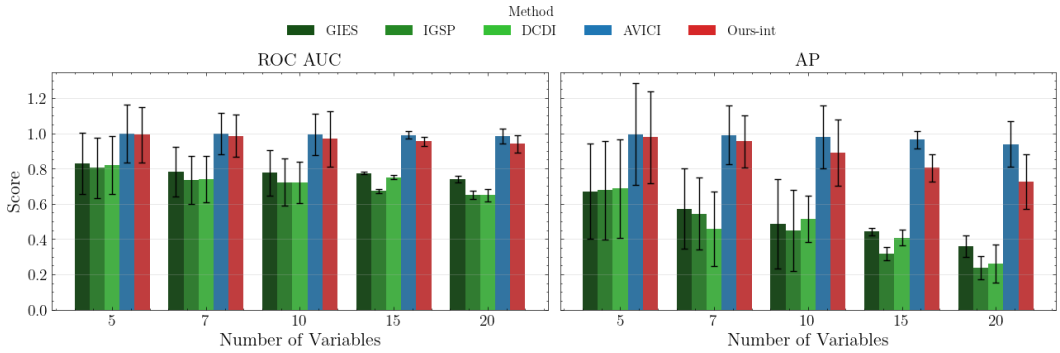

Figure 2: While we outperform statistical baselines (in green), we perform closely to AVICI on ROC AUC (left) in a stable manner, yet witness an increasing degradation in AP at scale (right).

understanding in comparison to the first and last layers. This aligns with interpretations about how models can define functional understanding in the middle layers, with layers towards the end being more adapted to the downstream tasks (Sia et al., 2024; Küken et al., 2025).

RQ3: How important is the encoder? We investigate whether TabPFN's encoder captures causal structure or if our performance is mainly attributed to the decoder. Figure 3 compares embeddings from four variants: the official pre-trained encoder (Optimal Weights), a randomly initialized encoder (Random Weights), embeddings before the encoder (Pre-encoder), and a fine-tuned version trained to perform worse on classification tasks (Worse Weights). Our results show that using weights and data embeddings from the pre-trained encoder improves performance in causal discovery, while degrading predictive performance (Worse Weights) also diminishes causal accuracy, suggesting that TabPFN's pre-training encodes feature interactions aligned with underlying causal relationships. Appendix D further shows that the decoder architecture has a comparably strong impact.

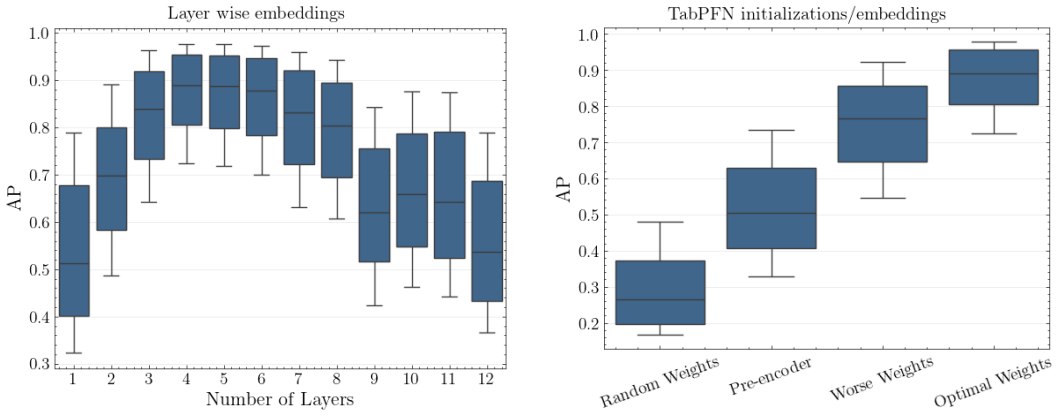

Figure 3: AP scores of our approach trained using different encoder/decoder layers (left) and different initializations/embeddings (right), showing that the middle layers and that embeddings from TabPFN's encoder of optimal weights encode better causal information.

5 Conclusion and future work

We show that TabPFN's embeddings contain causal information and that our adaptor framework outperforms traditional causal discovery algorithms when causal information is extracted from mid-range layers. This further promotes leveraging pre-trained tabular models for extracting causal structures, improving the interpretability of these models, and aiding in scientific discovery. Our framework can further help understand how tabular foundation models reason about different datasets, and provides a way to repurpose tabular foundation models for different downstream tasks.

Acknowledgments

This research was partially supported by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under SFB 1597 (SmallData), grant number 499552394. The authors acknowledge support by the state of Baden-Württemberg through bwHPC and the German Research Foundation (DFG) through grant INST 35/1597-1 FUGG. Frank Hutter acknowledges financial support by the Hector Foundation. Omar Swelam acknowledges support by the Konrad Zuse School of Excellence in Learning and Intelligent Systems (ELIZA) through the DAAD programme Konrad Zuse Schools of Excellence in Artificial Intelligence, sponsored by the Federal Ministry of Education and Research. The authors acknowledge support from ELLIS and ELIZA. Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union.

References

Barabási, A.-L. and Albert, R. Emergence of scaling in random networks. *science*, 286(5439): 509–512, 1999.

Brouillard, P., Lachapelle, S., Lacoste, A., Lacoste-Julien, S., and Drouin, A. Differentiable causal discovery from interventional data. In *Advances in Neural Information Processing Systems*, volume 33, pp. 21865–21877, 2020.

Chickering, D. M. Optimal structure identification with greedy search. *Journal of Machine Learning Research*, 2002.

Dhir, A., Ashman, M., Requeima, J., and van der Wilk, M. A meta-learning approach to bayesian causal discovery. In *The Thirteenth International Conference on Learning Representations*, 2025.

Feuer, B., Schirrmeister, R. T., Cherepanova, V., Hegde, C., Hutter, F., Goldblum, M., Cohen, N., and White, C. Tunetables: Context optimization for scalable prior-data fitted networks. *Advances in Neural Information Processing Systems*, 37:83430–83464, 2024.

Gilbert, E. N. Random plane networks. *Journal of the society for industrial and applied mathematics*, 9(4):533–543, 1961.

Hauser, A. and Bühlmann, P. Characterization and greedy learning of interventional markov equivalence classes of directed acyclic graphs. *The Journal of Machine Learning Research*, 13(1): 2409–2464, 2012.

Holland, P. W., Laskey, K. B., and Leinhardt, S. Stochastic blockmodels: First steps. *Social networks*, 5(2):109–137, 1983.

Hollmann, N., Müller, S., Eggensperger, K., and Hutter, F. Tabpfn: A transformer that solves small tabular classification problems in a second. *arXiv preprint arXiv:2207.01848*, 2022.

Hollmann, N., Müller, S., Purucker, L., Krishnakumar, A., Körfer, M., Hoo, S. B., Schirrmeister, R. T., and Hutter, F. Accurate predictions on small data with a tabular foundation model. *Nature*, 01 2025. doi: 10.1038/s41586-024-08328-6.

Ke, N. R., Chiappa, S., Wang, J. X., Bornschein, J., Goyal, A., Rey, M., Weber, T., Botvinick, M., Mozer, M. C., and Rezende, D. J. Learning to induce causal structure. In *International Conference on Learning Representations*, 2023.

Küken, J., Purucker, L., and Hutter, F. Early stopping tabular in-context learning. *arXiv preprint arXiv:2506.21387*, 2025.

Lee, H.-C., Danieletto, M., Miotto, R., Cherng, S. T., and Dudley, J. T. Scaling structural learning with NO-BEARS to infer causal transcriptome networks. In *Pacific Symposium on Biocomputing 2020*, pp. 391–402. World Scientific, 2019.

Lorch, L., Sussex, S., Rothfuss, J., Krause, A., and Schölkopf, B. Amortized inference for causal structure learning. *Advances in Neural Information Processing Systems*, 35:13104–13118, 2022.

Loshchilov, I. and Hutter, F. SGDR: Stochastic gradient descent with warm restarts. In *International Conference on Learning Representations*, 2017.

Loshchilov, I. and Hutter, F. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019.

Ma, Y., Frauen, D., Javurek, E., and Feuerriegel, S. Foundation models for causal inference via prior-data fitted networks. *arXiv preprint arXiv:2506.10914*, 2025.

Mokhtarian, E., Elahi, S., Akbari, S., and Kiyavash, N. Recursive causal discovery. *Journal of Machine Learning Research*, 26(61):1–65, 2025.

Robertson, J., Hollmann, N., Awad, N., and Hutter, F. Fairpfn: Transformers can do counterfactual fairness. *arXiv preprint arXiv:2407.05732*, 2024.

Robertson, J., Reuter, A., Guo, S., Hollmann, N., Hutter, F., and Schölkopf, B. Do-PFN: In-context learning for causal effect estimation. *arXiv preprint arXiv:2506.06039*, 2025.

Shimizu, S., Hoyer, P. O., Hyvärinen, A., Kerminen, A., and Jordan, M. A linear non-gaussian acyclic model for causal discovery. *Journal of Machine Learning Research*, 7(10), 2006.

Sia, S., Mueller, D., and Duh, K. Where does in-context learning happen in large language models? *Advances in Neural Information Processing Systems*, 37:32761–32786, 2024.

Spirtes, P., Glymour, C., and Scheines, R. Causation, prediction, and search. *MIT Press*, 2000.

Wang, Y., Solus, L., Yang, K., and Uhler, C. Permutation-based causal inference algorithms with interventions. *Advances in Neural Information Processing Systems*, 30, 2017.

Watts, D. J. and Strogatz, S. H. Collective dynamics of 'small-world'networks. *nature*, 393(6684): 440–442, 1998.

Zheng, X., Aragam, B., Ravikumar, P. K., and Xing, E. P. Dags with no tears: Continuous optimization for structure learning. *Advances in neural information processing systems*, 31, 2018.

A Architecture

Our architecture integrates three main components that interact sequentially to extract and model causal structure from tabular data. First, TabPFN's pre-trained embedding layer and encoder transform each value in a dataset into a contextualized embedding that captures relationships across samples and features. These embeddings are then processed by a decoder equipped with learnable causal tokens, which attend to the encoder outputs to identify potential causal directions. Finally, a DAG prediction module aggregates the causal token representations and produces pairwise edge probabilities, forming the estimated causal graph.

A.1 TabPFN encoder architecture and input preprocessing

Our encoder leverages the pre-trained TabPFNv2 (Hollmann et al., 2025) architecture, which now produces a dedicated representation for each feature value in each row in the input dataset.

Specifically, we embed the input data matrix $X \in \mathbb{R}^{n \times f}$ into a tensor $\mathcal{H} \in \mathbb{R}^{n \times f \times d}$, where d is the embedding dimension. This is accomplished by passing each entry X_{ij} through a TabPFN-learned projection layer that maps scalar cell values into d-dimensional vectors where $d = 192$.

To differentiate between interventional and observational samples, each feature in each sample is assigned a binary label indicating whether it is interventional or observational. This makes our data input shape different from the original TabPFN input shape, which is (n, f), to be $(n, f, 2)$ so we have two values for each feature in a datapoint, one indicating its real value and another indicating its interventional value. To get the per-feature representation, each 2 feature values are grouped into a single representation, so the input shape of $(n, f, 2)$ is then encoded by TabPFN's initial projection layer into (n, f, d) as needed. It is worth noting that given the causal sufficiency assumption, each feature corresponds to a node in the DAG generating the dataset.

Following the projection encoding layer, TabPFN's main encoder architecture uses a transformer encoder with a dual-attention mechanism that processes information along both the sample and feature dimensions via self-attention. After L Transformer layers, each sample token's state encodes information from other samples as well as from different features. The encoder's final output is $\mathcal{H}_{out} \in \mathbb{R}^{n \times f \times d}$. While TabPFN's original architecture has 12 layers ($L = 12$), we use $L = 4$ for our experiments, given the findings shown in Figure 3.

A.2 Causal tokens and decoder

Our main approach is dependent on having universal tokens that are tuned to capture the causal effect direction, which is inspired by TuneTables (Feuer et al., 2024). We initialize the causal tokens as learnable embeddings of the shape $(20, t \times d)$ where $t = 30$ and 20 is set as the maximum number of features we considered in our setup. For datasets with fewer than f features, we use only the first f embeddings, which are then reshaped from $(f, t \times d)$ to (t, f, d).

The learnable decoder performs cross-attention with the data tokens $\mathcal{H}_L \in \mathbb{R}^{n \times f \times d}$, obtained from layer L of TabPFN's frozen encoder, serving as the attention *keys* and *values*, while the causal tokens act as the *queries*. This design allows the causal tokens to selectively attend to relevant data representations and progressively integrate causal information across layers.

Structurally, the decoder mirrors the encoder's dual-attention architecture, where each layer contains two alternating multi-head attention blocks applied *across features* and *across samples*, interleaved with feed-forward sublayers. The key difference lies in the attention direction: instead of self-attending over data tokens, the decoder performs cross-attention from causal tokens to data tokens in both attention types. The number of layers and representational dimensionality are identical to those of the encoder to maintain architectural symmetry and stable information flow.

In our framework, the decoder is trained to encapsulate the causal relationships encoded in the data tokens into the causal tokens $\mathcal{Q}_L \in \mathbb{R}^{t \times f \times d}$. These causal tokens thus serve as compact, learnable summaries that aggregate structural dependencies across features, which are later decoded into the predicted adjacency matrix.

A.3 DAG prediction

To improve computational efficiency while preserving statistical richness, we aggregate the information across the t causal tokens into $k = 4$ representative tokens by applying four statistical operations, namely *max*, *min*, *mean*, and *std*. The resulting k tokens are concatenated along the representational dimension to form a matrix of shape $(f, k \times d)$ containing per-feature representations. Following Lorch et al. (2022), for each feature i ($i = 1, 2, \ldots, f$), we apply linear projections to obtain a parent representation $V_i \in \mathbb{R}^{k \times d}$ and a child representation $U_i \in \mathbb{R}^{k \times d}$. The pairwise adjacency score between features i and j is then computed as the dot product of their respective parent and child representations, followed by a sigmoid activation to yield the edge probability.

B Loss function

Binary cross-entropy loss: The binary cross-entropy loss formulation is:

$$\mathcal{L} = -\frac{1}{f(f-1)} \sum_{i \neq j} \left[A_{ij} \log \hat{A}_{ij} + (1 - A_{ij}) \log(1 - \hat{A}_{ij}) \right], \tag{1}$$

where f is the number of features and A_{ij} is the ground-truth adjacency matrix. The loss is weighted based on the number of edges in the graph to balance the learning of the model across different numbers of features.

Acyclicity constraint: For constraining the predicted adjacency matrix to be acyclic, the power iteration method is used to estimate the largest eigenvalue of the adjacency matrix to maintain numerical stability. Such constrained optimization follows a dual formulation for our model's learnable parameters through an augmented Lagrangian approach where dual variables track constraint violations, and the penalty weights are dynamically adjusted during training.

C Data generation

When sampling a dataset, we first sample the graph structure used to construct the directed acyclic graph (DAG). To generate the dataset values, one data generating function type, either linear or random Fourier feature (RFF), is then randomly selected and applied to define the functional mechanisms of all nodes.

C.1 Graph structures

The DAG generation process employs multiple graph generation mechanisms to ensure diverse causal topologies during training, mainly using five distinct graph generation processes. Figure 4 shows samples from the different graph structures listed below:

Erdős-Rényi Random Graphs: Each edge is sampled independently with a fixed probability, serving as a baseline topology.

Scale-Free Graphs: Incoming or outgoing edges of nodes are added to the previous node with probability proportional to $\deg(i)^{\alpha}$ (Barabási & Albert, 1999). This creates graphs with heavy-tailed degree distributions commonly observed in biological and social networks.

Watts-Strogatz Small-World Networks: These are k-dimensional lattices where edges are rewired globally to random nodes (Watts & Strogatz, 1998).

Stochastic Block Model: This model generates graphs with community structure by first partitioning nodes into random blocks, then setting inter-block edge probabilities lower than intra-block probabilities, capturing hierarchical structures found in complex systems (Holland et al., 1983).

Geometric Random Graphs: Nodes are randomly placed in a unit square, and edges are formed based on two-dimensional Euclidean distance below a threshold (Gilbert, 1961).

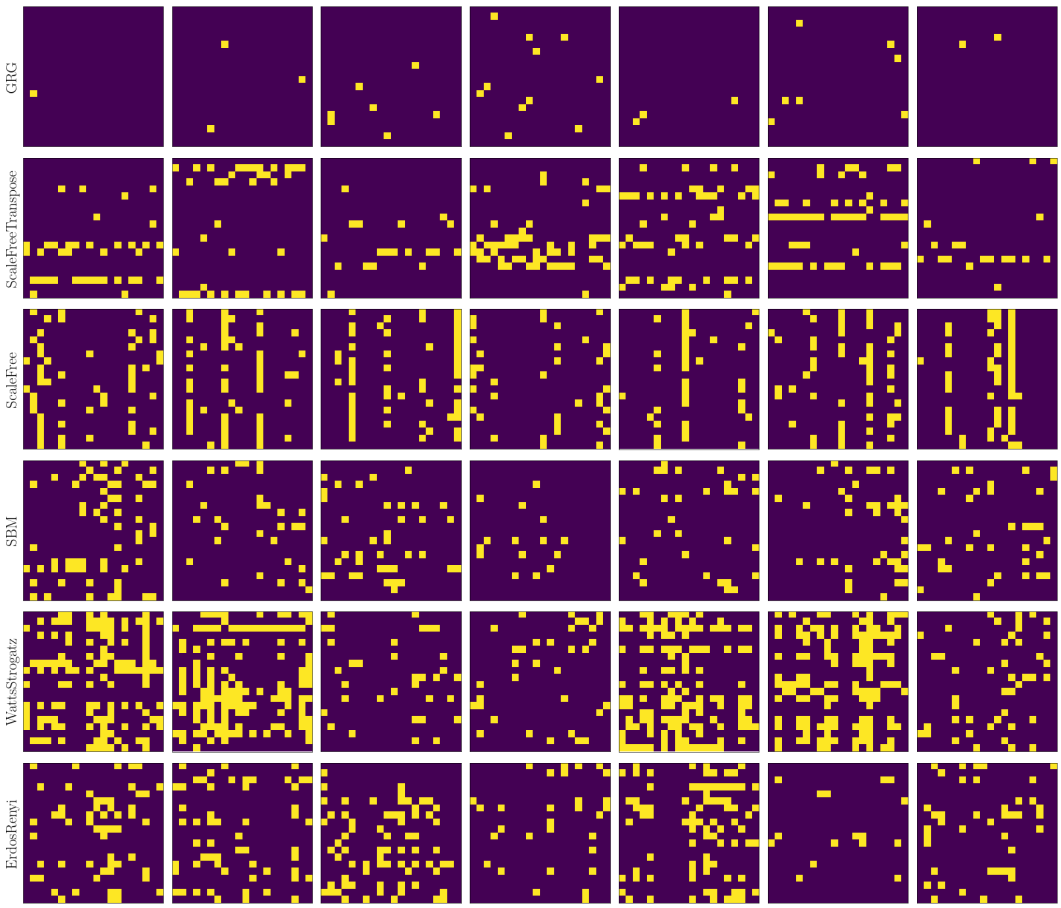

Figure 4: Samples of different graphs sampled from the graph structures. Each plot represents the adjacency matrix of a sampled DAG, where the yellow dots represent the edges.

C.2 Data generating functions

The relationships within the SCM, corresponding to a dataset, are defined such that each causal variable x_j is sampled given its parents $\boldsymbol{x}_{\text{pa}(j)}$ as

$$x_j \leftarrow f_j(\boldsymbol{x}_{\text{pa}(j)}, \epsilon_j) = f_j(\boldsymbol{x}_{\text{pa}(j)}) + h_j(\boldsymbol{x}_{\text{pa}(j)})\varepsilon_j \tag{2}$$

where the noise ε_j is additive. The data generating function f_j can be either linear or a random Fourier feature (RFF) function as an approximation for a Gaussian Process. The noise term can be sampled either from a Gaussian, Laplace, or Cauchy distribution. The specification of the parameter space for the graph structures, data generating functions, and noise scales follows a similar setup to Lorch et al. (2022), to have a consistent framework to benchmark against.

C.3 Interventions

The interventions in our generation process are single-variable interventions, where in each interventional data point, we intervene on a single variable. Such intervention is only performed on a subset of half of the available features. The intervention values are sampled from a uniform distribution.

C.4 Parameters of data generation

Table 1: A description of the data generation parameters used in our experiments. Graph structures are sampled with equal probability in all cases whenever specified within the domain of the distribution.

	Parameter	**Values**
Graph		
Erdős-Rényi	expected edges/node	$\in \{1, 2, 3\}$
Scale-free (in-degree)	edges/node	$\in \{1, 2, 3\}$
	attach. power α	$\in \{0.7, 1.0, 1.2, 1.5\}$
Scale-free (out-degree)	edges/node	$\in \{1, 2, 3\}$
	attach. power α	$\in \{0.7, 1.0, 1.2, 1.5\}$
Watts-Strogatz	lattice dim. k	$\in \{2, 3\}$
	rewire prob.	$\in \{0.2, 0.4\}$
Stochastic Block Model	expected edges/node	$\in \{1, 2, 3\}$
	blocks	$\in \{2, 5, 10\}$
	damp. inter-block prob.	$\in \{0.1\}$
Geometric Random Graphs	radius	$\in \{0.08, 0.1, 0.15\}$
Mechanism		
Linear function	weights $\boldsymbol{w}$	$\sim \text{Unif}_{\pm}(0.25, 4)$
	bias b	$\sim \text{Unif}(-3, 3)$
Random Fourier function	length scale ℓ	$\sim \text{Unif}(5, 12)$
	output scale c	$\sim \text{Unif}(8, 22)$
	bias b	$\sim \text{Unif}(-3, 3)$
Noise		
$\mathcal{N}(0, \sigma^2)$	σ	$\sim \text{Unif}(0.2, 2)$
Laplace$(0, \sigma^2)$	$\sigma^2(\boldsymbol{x}_{\text{pa}_j})$	$\sim p(h_{\text{rff}})$
Cauchy$(0, \sigma^2)$	$\sigma^2(\boldsymbol{x}_{\text{pa}_j})$	$\sim p(h_{\text{rff}})$
Interventions		
Target nodes	selection	random 50% of nodes
Intervention values	x_j	$\sim \text{Unif}_{\pm}(1, 5)$

Aliases:

- $\text{Unif}_{\pm}(a, b)$: uniform mixture of $\text{Unif}(a, b)$ and $\text{Unif}(-b, -a)$
- $p(h_{\text{rff}})$: distribution over heteroscedastic noise scale functions, induced by the squash function $h_{\text{rff}}(\boldsymbol{x}) = \log(1 + \exp(g_{\text{rff}}(\boldsymbol{x}))$ and random Fourier feature functions $g_{\text{rff}}(\boldsymbol{x})$.

D Ablations

D.1 Performance across decoder architectures

While we have seen how the encoder choice affects our performance, we wanted to inspect how the decoder also plays a role. Therefore, we introduce two decoder setups as shown in Figure 5 along with a "No Decoder" one:

- **No Decoder:** We do not use a decoder and pass the causal tokens into TabPFN's encoder as query tokens as if they are X_{test} data points. Thus, using the attention mechanism of TabPFN's encoder.
- **Standard Decoder:** We do cross attention with the attention source being data embeddings from the encoder's layer of choice, typically the 4th layer (our chosen approach).
- **Evolving Standard Decoder:** We do cross attention with the attention source in each decoder layer being data embeddings from the corresponding layer of the encoder, e.g., the 1st decoder layer attending to the output of the 1st encoder layer.

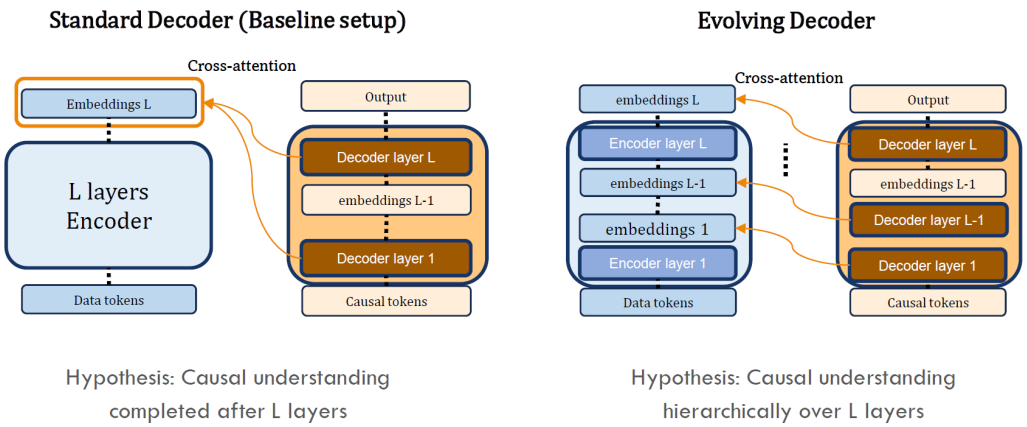

Figure 5: Different decoder architectures.

As shown in Figure 6, we have noticed how the performance of our approach varies across different decoder architectures, where having a learnable decoder plays a crucial role in the performance of our approach. The Standard Decoder setup witnesses the highest scores, while the No Decoder setup witnesses the lowest performance. This also indicates that the causal information can be considered completely evolved after the 4th layer of the encoder, not in a hierarchical manner.

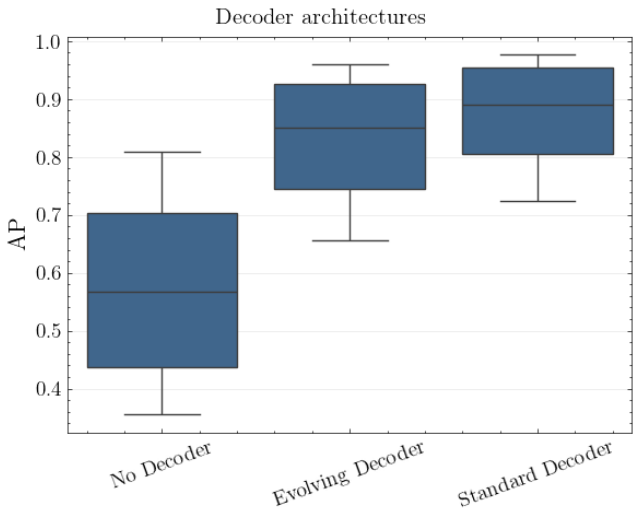

Figure 6: The presence of a decoder significantly improves the performance, showing that both the encoder's embeddings and the learnable decoder are essential for the causal information extraction.

D.2 Performance across graph structures

As shown before, we have noticed how the performance degrades across an increasing number of features in terms of AP scores. This further motivated us to analyze how such performance varies across different graph structures to understand what causes such degradation.

As shown below in Figure 7, the scaling issue is less pronounced in graph structures that have central nodes with a higher density of edges or sparse setups (namely, Scale-Free and GRG, respectively), while the variance is more pronounced in dense graph structures with a relatively larger number of edges distributed across different nodes (namely, Erodos-Renyi, SBM, and Watts-Strogatz), as shown in the samples in Figure 4. This can be attributed to the pre-training setup of TabPFN towards predictive tasks that aim at consolidating the information regarding a target variable, regardless of the rest of the feature interactions. This is further supported by the performance of AVICI over the different graph structures. Although AVICI witnesses a lower performance on Watts-Strogatz, the performance is relatively stable across other structures compared to our approach.

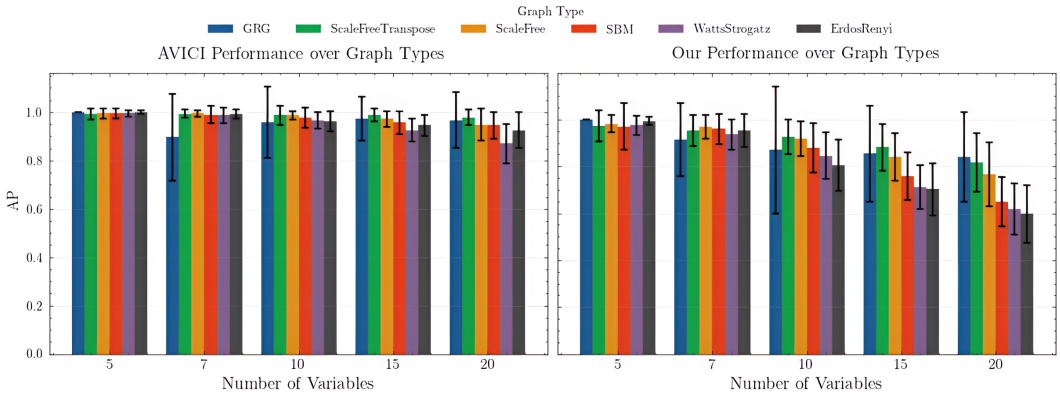

Figure 7: AP scores of our approach (right) compared to AVICI (left) for the different graph structures, showing how sensitive our approach is to the type of graph structures it witnesses at increasing scale.