Embedding-Based Markov Blanket Discovery for Robust Feature Selection Across Environments

Dhruv Bansal
SCAI
Arizona State University
Tempe, Arizona, USA
dbansa11@asu.edu

Sahajpreet Singh Khasria
SCAI
Arizona State University
Tempe, Arizona, USA
skhasria@asu.edu

1 Abstract

Causal feature selection through Markov Blanket (MB) discovery provides theoretical guarantees for optimal prediction while preserving model interpretability. However, identifying the MB without access to the underlying causal graph presents significant challenges, particularly when data exhibits distribution shift across environments. We present a meta-learning framework that combines TabPFN embeddings with neural MB predictors to perform simultaneous feature selection and regression across heterogeneous tasks. Our method trains separate multi-layer perceptron classifiers on TabPFN-extracted representations to predict binary MB masks, which subsequently filter features for downstream regression. Evaluated on 182 meta-training tasks and 46 held-out tasks spanning multiple data-generating processes and environmental conditions, our approach achieves a final score of 0.2071 (RMSE: 0.5078, Jaccard: 0.5921), demonstrating the feasibility of learning MB structure from embeddings without explicit causal discovery while outperforming the baseline by approximately 7.6%.

2 Introduction

Feature selection is fundamental to machine learning, directly impacting model performance, interpretability, and generalization. Traditional methods often select features based on correlation with the target variable, which can fail under distribution shift when spurious correlations change across environments (Pearl 1988). Causal feature selection addresses this limitation by identifying features with stable causal relationships to the target.

The Markov Blanket (MB) of a target variable provides a theoretically optimal feature set for prediction. Pearl (Pearl 1988) showed that the MB renders the target conditionally independent of all other variables, implying that knowing the MB is both sufficient and necessary for optimal prediction. For a target variable Y, the MB consists of its parents, children, and spouses (parents of children) in the causal graph. This theoretical foundation suggests that identifying the MB should yield robust predictions that generalize across environmental shifts.

Despite this theoretical support, discovering the MB in practice remains challenging. Standard approaches require causal discovery algorithms to first recover the full causal graph, then extract the MB. However, causal discovery methods are computationally expensive, require strong assumptions, and can be unreliable on finite samples (Tsamardinos and Aliferis 2003). This motivates the search for alternative approaches that can identify relevant causal features without explicit graph reconstruction.

We address this challenge by framing MB discovery as a meta-learning problem. Given a distribution over causal data-generating processes, we learn to predict MB masks directly from tabular data. Our approach leverages TabPFN (Hollmann et al. 2023), a transformer-based foundation model for tabular data, to extract rich representations that capture both predictive patterns and structural properties of datasets. These embeddings serve as inputs to learned MB predictors that generalize across tasks with varying feature dimensions and causal structures.

Main Contributions:

1. A practical pipeline for MB-based feature selection that bypasses explicit causal discovery
2. Demonstration that TabPFN embeddings encode sufficient information for cross-task MB prediction
3. Empirical validation across 228 tasks with diverse data-generating processes and environmental conditions

3 Related Works

Markov Blanket Theory. Pearl (Pearl 1988) formalized the concept of the Markov Blanket and proved its sufficiency for prediction. Tsamardinos and Aliferis

[0]Code available at: https://github.com/dhruvb26/CSE472-blanket-challenge

(Tsamardinos and Aliferis 2003) developed algorithms for MB discovery from observational data, establishing principled methods like IAMB for incremental MB learning (Tsamardinos et al. 2022). Recent work has shown that individual causal feature subsets (parents or children alone) can underperform non-causal baselines, emphasizing the importance of the complete MB structure.

TabPFN and Tabular Foundation Models. Hollmann et al. (Hollmann et al. 2023) introduced TabPFN, a transformer trained on synthetic datasets that performs in-context learning for tabular prediction. Unlike traditional models requiring task-specific training, TabPFN processes train-test pairs in a single forward pass, making predictions based on learned priors over data-generating processes. The model's embedding layer provides dense representations that capture both statistical patterns and structural properties of tabular data.

Meta-Learning for Tabular Data. Meta-learning frameworks learn to adapt quickly to new tasks by leveraging experience from related tasks (Finn et al. 2017). In the tabular domain, this typically involves learning good initializations or adaptation strategies. Our approach differs by learning a direct mapping from embeddings to structural properties (MB masks), treating MB discovery as a multi-label classification problem across tasks.

Causal Feature Selection. Traditional causal feature selection methods like the PC algorithm, GES, and NOTEARS first recover the causal graph, then extract relevant features. These methods achieve moderate accuracy with significant computational cost. Our embedding-based approach bypasses explicit graph recovery, directly predicting MB masks in a fraction of the time.

4 Method & Implementation

4.1 Model Design

Our approach consists of a three-stage pipeline that performs MB prediction followed by filtered regression. Each task $\mathcal{T}$ is represented by its support set (X_{train}, y_{train}) and query set (X_{test}, y_{test}) together with metadata such as feature masks, SCM parameters, and environment shifts.

Our pipeline processes tasks independently but uses a shared meta-trained set of models. We handle different feature dimensions by training separate TabPFN regressors, embedding generators, and MBMLP models for each feature size. For example, feature dimensions 19 and 9 are treated as separate groups, each with its own fine-tuned models.

4.2 Stage 1: TabPFN Fine-Tuning and Embedding Extraction

The primary technique in our pipeline is batched fine-tuning of the TabPFNRegressor (Prior Labs 2024). Using the built-in meta dataset collator, each batch corresponds to a full task that includes split preprocessing, differentiable forward passes, and updates to the model weights. This allows the regressor to adapt to the variety of training tasks instead of relying purely on pretraining.

For embedding extraction, we use TabPFNEmbedding to generate fold-based embeddings of each task. For each task with support set (X_{train}, y_{train}), we employ K-fold cross-validation:

1. Split X_{train} into K folds
2. For each fold k, train TabPFN on the remaining $K-1$ folds
3. Extract embeddings for the held-out fold
4. Concatenate embeddings across folds

These embeddings average over predictions and internal activations from the regressor, yielding a compact representation of the causal relationships in the data.

4.3 Stage 2: MB Prediction via Neural Networks

The meta-training dataset contains tasks with two distinct feature dimensionalities: tasks with 9 features and tasks with 19 features. Rather than using a single unified model with padding, we train separate MB predictors for each dimensionality.

For tasks with $d \in \{9, 19\}$ features, we define:

$$\text{MLP}_d : \mathbb{R}^E \to [0,1]^d \tag{1}$$

$$h_1 = \text{ReLU}(W_1 x + b_1) \tag{2}$$

$$h_2 = \text{ReLU}(W_2 h_1 + b_2) \tag{3}$$

$$\hat{m} = \sigma(W_3 h_2 + b_3) \tag{4}$$

where E is the embedding dimension, and σ is the sigmoid function. The output $\hat{m} \in [0,1]^d$ represents per-feature probabilities of MB membership.

Training Procedure. The MBMLP model performs multi-label binary classification to map embeddings to MB masks. BCEWithLogitsLoss is used during

training, and threshold tuning is carried out per feature dimension by evaluating several candidate thresholds on held-out tasks. After training the MBMLP model, we perform threshold tuning for each feature dimension by sweeping thresholds between 0.2 and 0.8, choosing the value that minimizes the combined RMSE and Jaccard-based score.

Inference. At test time, we extract embeddings for a new task, pass them through the appropriate MLP_d, and aggregate predictions via mean pooling followed by thresholding. If no features are selected (all probabilities below threshold), we employ a fallback strategy: select all features to ensure valid predictions.

4.4 Stage 3: Filtered Regression

Given predicted MB mask $\hat{m} \in \{0,1\}^d$, we filter both support and query sets:

$$X'_{train} = X_{train}[:, \hat{m} = 1] \tag{5}$$
$$X'_{test} = X_{test}[:, \hat{m} = 1] \tag{6}$$

We then use `clone_model_for_evaluation` to obtain clean, non-batched TabPFN models. The fine-tuned TabPFN is fitted on the filtered support set (X'_{train}, y_{train}) and used to generate predictions for the filtered query set X'_{test}. TabPFN's in-context learning capability allows it to adapt to the filtered feature space without requiring iterative optimization.

4.5 Implementation Details

Our entire pipeline is organized in the `/solution` directory with `main.py` as the entry point. Data loading from HuggingFace is handled in the `BeyondTheBlanket` class initialization.

Key files:

- `finetuner.py` - TabPFN fine-tuning logic
- `generator.py` - Embedding generation
- `mbmlp.py` - Markov Blanket MLP predictor

5 Experimental Setup

5.1 Dataset and Task Distribution

We use the CSE472-blanket-challenge/final-dataset from Hugging Face, which contains tasks generated from a hierarchical causal data-generating process. Each task $\mathcal{T} = (G, \text{SCM}, E)$ is defined by a directed acyclic graph G, a structural causal model specifying functional relationships, and an environment $E \in \{\text{IID}, \text{covariate shift}, \text{label shift}\}$.

The meta-training set (develop) contains 182 tasks with the following distribution:

- **Feature dimensions:** 87 tasks with 9 features, 95 tasks with 19 features
- **Environments:** IID, label shift, covariate shift
- **SCM types:** Linear and nonlinear
- **Sample sizes:** 400 training samples, 100 test samples per task

The held-out set (submit) contains 46 tasks with ground truth MB masks and test labels withheld for final evaluation.

5.2 Training Configuration

For meta-training, we grouped tasks by their feature dimensionality. Within each feature dimension group, we held out 10 percent of the tasks for validation and threshold tuning, and used the remaining 90 percent for fine-tuning the TabPFN regressor and training the MBMLP network. Hyperparameter configurations for all pipeline components are provided in Table~2 in the Appendix.

5.3 Computational Resources

All experiments were conducted on ASU Sol HPC cluster using A100 GPUs (40GB VRAM). Total pipeline runtime was approximately 20 minutes including fine-tuning, embedding extraction, MBMLP training, and threshold tuning. Submit prediction uses cloned evaluation-mode TabPFN models and is comparatively lightweight.

6 Results

6.1 Main Quantitative Results

The results were calculated by running baseline tests and the final model on held-out tasks (~10% from the develop set).

The baseline delivers an average RMSE of 0.4878, average Jaccard of 0.5504, and combined score of 0.2193. Our pipeline delivers an average RMSE of 0.5078, average Jaccard of 0.5921, and an aggregate final score of **0.2071**.

Dim	Model	RMSE	Jaccard	Score
19	Baseline	**0.5071**	0.5088	0.2491
19	Final	0.5258	**0.5625**	**0.2301**
9	Baseline	**0.4661**	0.5972	0.1878
9	Final	0.4876	**0.6255**	**0.1826**

Table 1: Performance comparison by feature dimension

The final model produces a higher Jaccard score at the cost of slightly elevated RMSE. **The final model achieves approximately 5.6% overall improvement over the baseline** (7.6% for dimension 19, 2.8% for dimension 9).

6.2 Qualitative Analysis

The graph's complexity with entangled paths and dispersed blanket nodes makes MB prediction a particularly challenging problem. Beyond individual examples, a broader inspection of tasks reveals that MB prediction quality is closely tied to the structural clarity of the underlying causal graph. Tasks with well-separated parent and child relationships tend to produce higher Jaccard values, even when the regression RMSE remains moderate.

Figure~1 shows the causal graph for task `data_e963d7d6`, which achieved high Jaccard score (0.9) despite moderate RMSE (0.7790). The causal graph structure is relatively clear, with well-separated relationships contributing to accurate MB prediction. Figure~2 shows the causal graph for task `data_be3555b2`, which achieved lower Jaccard score (0.3) with RMSE 0.8202, due to more complex graph structure with entangled paths.

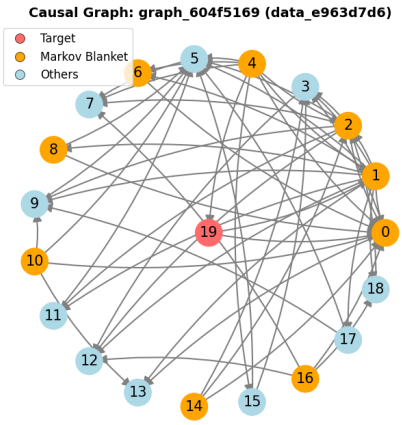

Figure 1: Causal graph for `data_e963d7d6` (RMSE: 0.7790, Jaccard: 0.9)

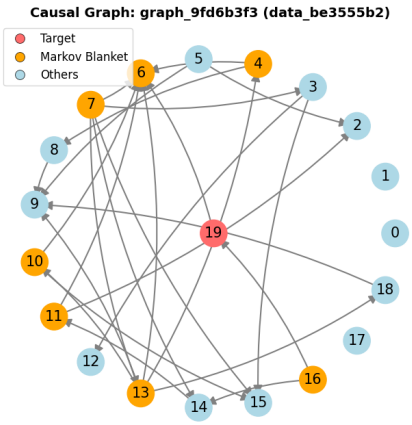

Figure 2: Causal graph for `data_be3555b2` (RMSE: 0.8202, Jaccard: 0.3)

7 Analysis & Insights

7.1 What Worked and Why

Task-structured training: For each supported feature dimension, develop tasks are split into a "support" subset for fine-tuning TabPFN and training the MB network, and a "query" subset for threshold tuning and evaluation, mirroring the benchmark's support/query structure.

Causal/MB signal: TabPFN embeddings for each task feed an MB-specific MLP that predicts Markov Blanket membership per feature dimension. Averaging per-task MB probabilities and thresholding keeps only features most likely in the causal Markov Blanket before running the regression fine-tune.

Joint score tuning: Held-out tasks drive threshold selection by minimizing a combined objective (RMSE $\times$ (1 $-$ Jaccard)), directly balancing predictive accuracy with MB fidelity and exploiting causal sparsity assumptions.

7.2 Failure Modes and Limitations

Sparse tasks per dimension: When few develop tasks exist for a feature dimension, the support split and MB training become data-starved, making the MB classifier and threshold tuning noisy.

MB instability: Thresholding can yield empty blankets, triggering the fallback to all features. Frequent fallback indicates the MB model cannot distinguish relevant features—often on high-dimensional or highly correlated tasks.

Distribution shift: The MB network and thresholds are tuned on develop held-outs; submit tasks with substantially different causal structure may see degraded blanket accuracy and downstream RMSE because the learned thresholds no longer align.

8 Future Work

8.1 Architectural Improvements

Unified Multi-Task Model. Our current approach trains separate MLP predictors for each feature dimension. A unified architecture using adaptive pooling or padding could leverage shared structure across dimensions while maintaining specialized capacity. Graph neural networks such as DAG-GNN (Yu et al. 2019) could model feature dependencies explicitly, potentially improving MB prediction for features with complex interdependencies.

Attention-Based Aggregation. We currently aggregate sample-level predictions via mean pooling. An attention mechanism could learn to weight samples based on their informativeness, potentially improving robustness to outliers or low-quality training examples.

8.2 Training Enhancements

TabPFN Fine-Tuning Extensions. TabPFN 2.5 supports fine-tuning via batched training mode. Rather than using zero-shot predictions, deeper fine-tuning on develop tasks could adapt the model to the specific distribution of causal data-generating processes, improving both embedding quality and downstream regression performance.

Uncertainty-Aware Selection. Our current threshold of 0.5 for MB membership is fixed. A learned or adaptive threshold based on prediction confidence could improve precision-recall trade-offs. Incorporating established causal discovery ideas like IAMB (Tsamardinos et al. 2022) would also provide a more principled signal for selecting parent and spouse variables.

8.3 Evaluation Extensions

Per-Environment Analysis. Our validation metric averages across all environments. Stratified evaluation would reveal whether performance degrades differentially under distribution shift, informing environment-specific adaptations.

Sample Efficiency Studies. Investigating performance as a function of support set size would characterize the method's data efficiency. TabPFN's in-context learning may enable reasonable performance with very few support examples.

9 Appendix

9.1 Hyperparameters

Component	Parameter	Value
TabPFN (fine-tuning)	epochs	1
TabPFN (fine-tuning)	learning_rate	1×10^{-6}
TabPFN (fine-tuning)	max_samples	10,000
TabPFN (fine-tuning)	val_fraction	0.1
MLP Predictor	hidden_dims	[256, 128]
MLP Predictor	learning_rate	1×10^{-3}
MLP Predictor	epochs	40
MLP Predictor	batch_size	512
MLP Predictor	val_fraction	0.2
TabPFN (regression)	n_estimators	24

Table 2: Hyperparameter configuration for all pipeline components

9.1.1 Explicit (Exposed through config.yaml)

Hyperparameter	Description
`supported_feature_dims`	Feature dimensions (e.g., 9, 19)
`holdout_fraction`	Fraction held out for evaluation

Table 3: General Configuration

Hyperparameter	Description
`tabpfn.epochs`	Fine-tuning epochs
`tabpfn.lr`	Learning rate (Adam)
`tabpfn.max_samples`	Max samples per dataset
`tabpfn.val_fraction`	Internal validation fraction
`tabpfn.model_path`	Pretrained model checkpoint

Table 4: TabPFN Fine-tuning Configuration

Hyperparameter	Description
`mb_mlp.epochs`	Training epochs
`mb_mlp.lr`	Learning rate
`mb_mlp.batch_size`	Training batch size
`mb_mlp.hidden_sizes`	Hidden layer widths
`mb_mlp.val_fraction`	Validation fraction
`mb_mlp.n_fold`	Folds for embedding extraction

Table 5: MBMLP Configuration

Hyperparameter	Description
`threshold_tuning.min`	Min threshold in sweep
`threshold_tuning.max`	Max threshold in sweep
`threshold_tuning.num_thresholds`	Grid search values

Table 6: Threshold Tuning Configuration

9.1.2 Implicit Hyperparameters

Component	Hyperparameter	Reason
TabPFN	`n_estimators = 24`	Ensemble size
MBMLP	`activation = ReLU`	Expressiveness
MBMLP	`loss = BCEWithLogitsLoss`	Training objective
MBMLP	`optimizer = Adam`	Learning trajectory
MBMLP	`betas = (0.9, 0.999)`	Momentum
Threshold	`rule = >=`	Mask selection
Threshold	`aggregation = mean`	MB selection
Threshold	metric = RMSE×(1-Jacc)	Best threshold

Table 7: Implicit Hyperparameters

9.2 Training Loss on MBMLP

Training loss curves for MBMLP models during meta-training:

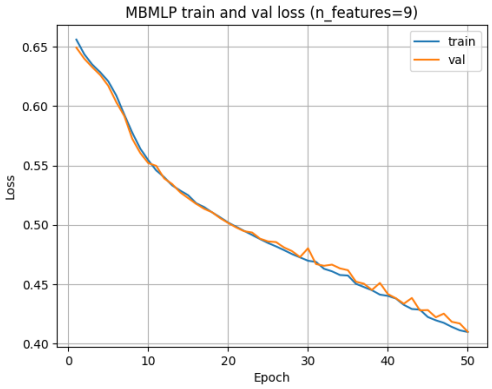

Figure 3: MBMLP training loss (9 features)

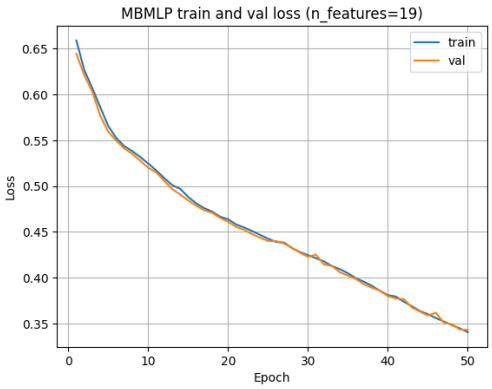

Figure 4: MBMLP training loss (19 features)

References

Finn, Chelsea, Pieter Abbeel, and Sergey Levine. 2017. "Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks." *Proceedings of the 34th International Conference on Machine Learning*, ICML'17, 1126–35. https://proceedings.mlr.press/v70/finn17a/finn17a.pdf.

Hollmann, Noah, Samuel Müller, Katharina Eggensperger, and Frank Hutter. 2023. "TabPFN: A Transformer That Solves Small Tabular Classification Problems in a Second." *arXiv Preprint arXiv:2207.01848*.

Pearl, Judea. 1988. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers.

Prior Labs. 2024. *Prior Labs Documentation*. Https://docs.priorlabs.ai/overview.

Tsamardinos, Ioannis, and Constantin F. Aliferis. 2003. "Towards Principled Feature Selection: Relevancy, Filters and Wrappers." *Proceedings of the Ninth International Workshop on Artificial Intelligence and Statistics*.

Tsamardinos, Ioannis, Constantin Aliferis, and Alexander Statnikov. 2022. "The Incremental Association Markov Blanket Algorithm." *Algorithms* 15 (4). https://www.mdpi.com/1999-4893/15/4/105.

Yu, Yue, Jianfei Gao, and Song-Chun Zhu. 2019. "DAG-GNN: DAG Structure Learning with Graph Neural Networks." *Proceedings of the 36th International Conference on Machine Learning*, ICML'19. http://proceedings.mlr.press/v97/yu19a/yu19a.pdf.