POTENCYGNN: AMORTIZED EPISTEMIC POTENCY WITH DUAL-FEASIBLE CERTIFICATES

Anonymous authors
Paper under double-blind review

ABSTRACT

Auxiliary observations can reduce causal uncertainty under latent confounding even when effects remain only partially identified. We formalize this gain through realized observational epistemic potency: the reduction in sharp identified-set width obtained when moving from the treatment–outcome marginal distribution to the full observational distribution while keeping the causal graph, query, and model class fixed. Exact computation requires four response-function linear programs and becomes rapidly expensive as parent counts grow. We introduce PotencyGNN, a query-conditioned relational graph neural network with separate parent, child, and bidirected message-passing channels, together with a dual-certificate network that provides certified lower bounds on potency. On screened core-disjoint three-to-five- and exact-six-variable benchmarks, PotencyGNN attains held-out $R^2 = 0.829$ and 0.779, remaining competitive with the evaluated graph transformer and outperforming the other baselines. Under matched CPU inputs, PotencyGNN inference is $8.22\times$ faster than cached exact evaluation for individual queries and achieves $26.17\times$ higher batched throughput. Across $79{,}200$ certificate evaluations, all satisfy the numerical validity criteria, with 43.4% yielding certified potency above the reporting threshold.

1 INTRODUCTION

Under latent confounding, a causal query may remain only partially identified even when the graph and observational distribution are known: its sharp identified set contains all values attainable by compatible structural causal models (Manski, 1990; Balke & Pearl, 1994; 1997). Auxiliary observed variables can tighten this set by ruling out compatible models. Epistemic potency quantifies guaranteed tightening before experimental outcomes are observed (Maringgele & Etesami, 2026); here, we study a realized observational variant for evidence already available. The base regime imposes the $P(X, Y)$ cell constraints and the full regime imposes the $P(V)$ constraints; both fix the graph, query, model class, response-function space, and outside-clique conditionals from $P(V)$. Potency is the resulting reduction in identified-set width and measures identification uncertainty rather than effect magnitude or sampling uncertainty.

Exact evaluation is costly. In the binary response-function formulation, each potency label requires four linear programs—lower and upper bounds under the base and full regimes—over a joint response-type space that grows doubly exponentially with observed parent counts. These programs must be solved again for every new instance. Specialized methods can reduce exact-computation cost (Shridharan & Iyengar, 2023a;b), and some problem classes admit analytic solutions; we instead amortize computation by learning from exact offline labels.

PotencyGNN combines query-conditioned observational summaries with directed and bidirected graph structure because tightening depends on both the observed distribution and each variable's relation to the treatment–outcome query. Since point prediction alone cannot certify that tightening occurs, a separate dual-certificate network provides conservative potency lower bounds through dual-feasibility repair and exact base bounds.

Our contributions are fourfold. First, we define realized observational epistemic potency as the reduction in sharp-bound width obtained from auxiliary observational variables and study its amortized prediction. Second, we introduce *PotencyGNN*, a query-conditioned relational surrogate over observational summaries, directed and bidirected structure, and the ordered treatment–outcome

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.

query. Third, we adapt learned dual prediction and deterministic feasibility repair to the full-evidence causal programs, yielding certified potency lower bounds in exact arithmetic. Finally, across core-disjoint benchmarks, we evaluate generalization, structural coverage, efficiency, certificate validity, and empirical transfer; PotencyGNN is competitive with strong graph baselines, accelerates repeated exact evaluation, supports certified screening, and benefits more from broader causal-core coverage than repeated sampling within fewer cores.

2 Related work

Partial identification and epistemic potency. Partial identification characterizes causal effects compatible with observed data and maintained assumptions (Manski, 1990). For discrete SCMs, response-function representations turn sharp bounding into optimization over latent response types (Balke & Pearl, 1994; 1997), with later work improving computational scalability (Shridharan & Iyengar, 2023a;b; Arroyo et al., 2025). Epistemic potency has been used to score experiments by the worst-case tightening guaranteed before outcomes are observed (Maringgele & Etesami, 2026), while related design work targets identifiability (Akbari et al., 2025). In contrast, we study the realized tightening already provided by auxiliary observational variables and amortize its exact evaluation.

Amortized causal inference. Learning-based causal inference has been amortized for structure discovery (Lorch et al., 2022) and effect estimation under ignorability (Balazadeh Meresht et al., 2026). Recent work also learns or approximates partially identified bounds using implicit generative models, in-context prediction, or prior-fitted mappings (Balazadeh Meresht et al., 2022; Tan et al., 2024; Meresht et al., 2026; Bellot & Dhir, 2026). Rather than predicting an identified set or causal effect, PotencyGNN predicts the reduction in width between two nested sharp identified sets for the same graph and ordered query.

Relation-aware graph learning. Relational GNNs parameterize message passing by edge type (Schlichtkrull et al., 2018; Battaglia et al., 2018), while direction-aware architectures distinguish incoming and outgoing relations (Rossi et al., 2024). PotencyGNN adapts these ideas to ADMGs through separate parent, child, and bidirected channels, query-conditioned observational summaries, and an ordered treatment–outcome readout. Unlike generic graph encoders, the architecture is designed around the relational roles relevant to causal-bound tightening.

Learned optimization and certification. Learned optimization can predict feasible solutions, accelerate dual optimization, or construct valid objective bounds through feasibility repair (Donti et al., 2021; Dinitz et al., 2021; Tanneau & Van Hentenryck, 2024). Our separate certificate network instead predicts dual variables for the full-evidence causal bounding programs; deterministic repair enforces dual feasibility, and intersection with exact base bounds yields a conservative potency lower bound. The certificate therefore guarantees tightening under the stated causal model, rather than certifying PotencyGNN's prediction error or providing statistical coverage.

3 Problem setup

Causal model and graph. Let $V = \{V_1, \dots, V_n\}$ denote observed binary variables, including treatment X, outcome Y, and auxiliary variables $V \setminus \{X, Y\}$. A structural causal model (SCM) $\mathcal{M} = (\boldsymbol{f}, P_U)$ specifies assignments $V_i = f_i(V_{\mathrm{pa}_G(V_i)}, U_i)$, where the exogenous variables U_i have joint distribution P_U; dependence among them can induce latent confounding. Its acyclic directed mixed graph (ADMG) $G = (V, E_{\to}, E_{\leftrightarrow})$ represents direct causal relations through directed edges and latent dependence through bidirected edges (Richardson, 2003); the directed part is acyclic, and directed and bidirected edges may coexist as a bow. Bidirected edges form a single clique $C \subseteq V$ containing X, Y, with no bidirected edges outside C. For partial identification, we restrict attention to a response-function class with $P_U = P_{U_C} \prod_{V_i \notin C} P_{U_i}$ and unrestricted P_{U_C}, where $U_C = (U_i : V_i \in C)$. Thus, exogenous variables outside C are mutually independent and independent of U_C, while the corresponding conditional distributions are fixed from $P(V)$. All sharpness claims apply to this restricted SCM class. We write $\mathrm{pa}_G(V_i)$ and $\mathrm{an}_G(V_i)$ for observed parents and

strict directed ancestors. Unless stated otherwise, $P(V)$ denotes the population observational distribution, while empirical evaluations use the corresponding empirical distribution $\hat{P}(V)$. Finally, every parent configuration of each $V_i \notin C$ is required to have positive probability under $P(V)$; we call this *Assumption S (outside-clique parent support)*. Appendix G states it formally and explains why it is required for the implemented sharp bounds.

Causal query and epistemic potency. For the ordered treatment–outcome pair (X, Y), we consider $Q(\mathcal{M}) = P_{\mathcal{M}}(Y = 1 \mid \mathrm{do}(X = 1))$, where $P_{\mathcal{M}}$ is the distribution induced by $\mathcal{M}$ and $\mathrm{do}(X = 1)$ sets $X = 1$ while leaving the other structural assignments unchanged. For $s \in \{\mathrm{base}, \mathrm{full}\}$, let $\mathfrak{M}_s$ be the set of SCMs in the restricted class above that satisfy the observational constraints of regime s. If nonempty, the attainable query values form the sharp identified set $\mathcal{I}_s(Q) = \{Q(\mathcal{M}) : \mathcal{M} \in \mathfrak{M}_s\} = [L_s, U_s]$, where L_s and U_s are its lower and upper bounds and $w_s = U_s - L_s$ is its width (Manski, 1990; Balke & Pearl, 1997); $w_s = 0$ denotes point identification. The regimes share the graph, query, SCM restrictions, and outside-C conditionals from $P(V)$. Base imposes the $P(X, Y)$ cell constraints; full imposes the $P(V)$ constraints, so $\mathcal{I}\mathrm{full}(Q) \subseteq \mathcal{I}\mathrm{base}(Q)$ and $w_{\mathrm{full}} \leq w_{\mathrm{base}}$. Potency therefore isolates tightening from adding the remaining $P(V)$ cell constraints conditional on the shared kernels. Diagnostics in Appendix A found unchanged base endpoints when these kernels were varied or freed, but we do not assume this invariance generally. With $P = P(V)$, we define epistemic potency as

$$\tau(G, P, Q) = w_{\mathrm{base}} - w_{\mathrm{full}} = (U_{\mathrm{base}} - L_{\mathrm{base}}) - (U_{\mathrm{full}} - L_{\mathrm{full}}).$$

Thus $0 \leq \tau(G, P, Q) \leq w_{\mathrm{base}}$. Zero potency means the auxiliary variables do not further reduce the identified-set width, while positive potency need not imply point identification. Potency measures neither causal effect magnitude nor sampling uncertainty.

Exact computation. We represent the restricted binary SCM class using deterministic response functions (Balke & Pearl, 1994; 1997). Let $k_i = |\mathrm{pa}_G(V_i)|$. For $V_i \in C$, a response type specifies its output for each of the 2^{k_i} parent configurations, yielding $2^{2^{k_i}}$ possible types. A joint response type selects one response type for each variable in C, and q denotes a probability distribution over the resulting joint types. For $V_i \notin C$, we instead fix $P(V_i \mid V_{\mathrm{pa}_G(V_i)})$ from $P(V)$, or $P(V_i)$ when V_i has no parents. These conditional distributions are shared by both regimes and are not optimization variables. The implementation assigns $1/2$ to undefined conditionals at zero-probability parent configurations, but Assumption S excludes such cases. For regime s, the feasible response-type distributions are $\mathcal{F}_s = \{q \geq 0 : \mathbf{1}^\top q = 1,\ A_s q = b_s\}$, where b_s contains the observational probabilities required by regime s, and each column of A_s gives one joint type's contribution to those probabilities. Section 4.1 incorporates the normalization constraint into A_{full}. Let $c_j \in [0, 1]$ denote the causal-query value for joint type j, averaging over variables outside C using their fixed conditional distributions. Under Assumption S, whenever $\mathcal{F}_s \neq \varnothing$, $\mathcal{I}_s(Q) = \{c^\top q : q \in \mathcal{F}_s\}$, so $L_s = \min_{q \in \mathcal{F}_s} c^\top q$ and $U_s = \max_{q \in \mathcal{F}_s} c^\top q$. Because $\mathcal{F}_s$ is convex and compact and the objective is linear, all intermediate values are attained; hence these bounds, and therefore w_s and τ, are sharp for the restricted SCM class. Computing one exact potency label requires four linear programs over $K = \prod_{V_i \in C} 2^{2^{k_i}}$ variables. Appendix A gives a complete example.

4 POTENCYGNN: AMORTIZED PREDICTION OF EPISTEMIC POTENCY

PotencyGNN amortizes exact potency evaluation with a query-conditioned surrogate $\hat{\tau} = f_\theta\big(G, P(V), X, Y\big)$. Rather than consuming $P(V)$ directly, f_θ constructs observational summaries from $P(V)$ and attaches them to the corresponding nodes.

Query-conditioned evidence representation. To avoid the ordering dependence and exponential dimension of a flattened $P(V)$, PotencyGNN assigns each node V_i the evidence profile

$$\mathbf{x}_i = [\, r_i^X, r_i^Y, r_i^C, \mu_i, \Delta_i^X, \Delta_i^Y, \rho_i^{00}, \rho_i^{01}, \rho_i^{10}, \rho_i^{11}, d_i^{\mathrm{in}}, d_i^{\mathrm{out}} \,] \in \mathbb{R}^{12}. \tag{1}$$

The indicators r_i^X, r_i^Y, r_i^C mark whether V_i is X, is Y, or lies in C, thereby conditioning a shared predictor on the ordered query; $\mu_i = P(V_i = 1)$ is the node marginal. The associations $\Delta_i^T = P(T = 1 \mid V_i = 1) - P(T = 1 \mid V_i = 0)$, $T \in \{X, Y\}$, set to zero when $V_i = T$, are observational rather than causal contrasts. The query-cell prevalences $\rho_i^{xy} = P(V_i = 1 \mid X =$

$x, Y = y)$, $(x, y) \in \{0, 1\}^2$, capture variation not generally recoverable from the marginal and pairwise associations alone, and $d_i^{\text{in}}, d_i^{\text{out}}$ are directed in- and out-degrees; full connectivity reaches the relational encoder separately. Since these node profiles contain neither query-cell masses nor problem scale, we add an instance-level context vector:

$$\mathbf{g} = [\underbrace{\pi_{00}, \pi_{01}, \pi_{10}, \pi_{11}}_{\text{query margin}}, \underbrace{n, |E_{\to}|, |C|, \log_2 K}_{\text{problem scale}}], \qquad \pi_{xy} = P(X = x, Y = y). \tag{2}$$

The π_{xy} entries supply the base regime's query-margin constraints, recording cell masses rather than within-cell prevalences ρ_i^{xy}; K is the number of joint response types. These observational summaries are not lossless: they do not explicitly preserve arbitrary higher-order interactions among auxiliary variables. Appendix A.1 provides the full feature specification and conventions for undefined conditionals.

Relation-specific message passing. Node-aligned profiles summarize each variable, but their implications depend on its ADMG position relative to the query. PotencyGNN therefore uses separate parent, child, and bidirected channels together with a self channel, preserving relational roles that a single undirected neighborhood would discard. Each profile is encoded as $\mathbf{h}_i^{(0)} = \text{ReLU}(W_{\text{enc}}\mathbf{x}_i + b_{\text{enc}}) \in \mathbb{R}^d$, where d is the hidden dimension. At layer ℓ, the model separately aggregates representations from directed parents, directed children, and bidirected neighbors:

$$\mathbf{m}_{i,\text{pa}}^{(\ell)} = \sum_{j:V_j \to V_i} \mathbf{h}_j^{(\ell)}, \qquad \mathbf{m}_{i,\text{ch}}^{(\ell)} = \sum_{j:V_i \to V_j} \mathbf{h}_j^{(\ell)}, \qquad \mathbf{m}_{i,\leftrightarrow}^{(\ell)} = \sum_{j:V_j \leftrightarrow V_i} \mathbf{h}_j^{(\ell)}, \tag{3}$$

and combines them with the node's current state:

$$\mathbf{h}_i^{(\ell+1)} = \text{ReLU}\left(\Phi_{\text{self}}^{(\ell)}(\mathbf{h}_i^{(\ell)}) + \sum_{t \in \{\text{pa,ch,}\leftrightarrow\}} \Phi_t^{(\ell)}(\mathbf{m}_{i,t}^{(\ell)})\right). \tag{4}$$

Each affine map is applied after aggregation, shared across nodes, and parameterized separately across channels and layers, so contributions depend on relation type; $\Phi_{\text{self}}^{(\ell)}$ is learned rather than an identity residual connection. Child-to-parent message flow acts on representations without reversing causal direction, while a neighbor connected by multiple edge types, as in a bow, contributes through each applicable channel. One layer reaches immediate neighbors, so stacking R layers propagates evidence from more distant nodes.

Ordered-query readout and late global-context fusion. After R relation-specific message-passing layers, the readout combines a graph-wide mean, separate treatment and outcome states, and global context for the ordered query (X, Y):

$$\bar{\mathbf{h}} = \frac{1}{|V|} \sum_{V_i \in V} \mathbf{h}_i^{(R)}, \qquad \mathbf{z} = [\,\bar{\mathbf{h}} \,\|\, \mathbf{h}_X^{(R)} \,\|\, \mathbf{h}_Y^{(R)} \,\|\, \mathbf{g}\,] \in \mathbb{R}^{3d+8}, \tag{5}$$

where $\|$ denotes concatenation. The mean $\bar{\mathbf{h}}$ provides a permutation-invariant graph summary, while the ordered treatment and outcome states preserve their distinct roles and allow sensitivity to query order. Predictions remain invariant to consistent relabeling of nodes, features, relations, and query indices. Fusing $\mathbf{g}$ only at readout supplies the query margin and problem scale directly rather than through message passing, although the node features may contain overlapping information. A prediction head with two ReLU hidden layers and a linear scalar output maps $\mathbf{z}$ to the potency estimate: $\mathbf{a}_1 = \text{ReLU}(W_1\mathbf{z} + b_1)$, $\mathbf{a}_2 = \text{ReLU}(W_2\mathbf{a}_1 + b_2)$, and $\hat{\tau} = W_3\mathbf{a}_2 + b_3$. For M training instances with exact potency labels $\{\tau_j\}_{j=1}^M$, we minimize the mean squared error

$$\mathcal{L}(\theta) = \frac{1}{M} \sum_{j=1}^{M} (\hat{\tau}_j - \tau_j)^2, \tag{6}$$

where each target is computed from the four exact bounding programs of Section 3.

4.1 DUAL-FEASIBLE CERTIFICATES FOR EPISTEMIC POTENCY

A point prediction may overstate identification gain, so we add a complementary lower certificate whose validity follows from deterministic dual-feasibility repair rather than prediction accuracy. The construction encloses the full-regime interval with repaired dual bounds, intersects that enclosure with the base interval, and subtracts its width from the base width.

For the full-evidence programs, write $A = A_{\text{full}} \in \mathbb{R}^{m\times K}$ and $b = b_{\text{full}} \in \mathbb{R}^m$, stacking the normalization row $\mathbf{1}^\top q = 1$ with the full-regime constraints so that $Aq = b$ contains all m equalities; Section 3 states normalization separately. Each column of A corresponds to one response type, with query contribution c_j. For feasible, bounded programs, strong duality gives

$$L_{\text{full}} = \min_{q\geq 0:\, Aq=b} c^\top q = \max_{y_L:\, A^\top y_L \leq c} b^\top y_L,\; U_{\text{full}} \;= \max_{q\geq 0:\, Aq=b} c^\top q = \min_{y_U:\, A^\top y_U \geq c} b^\top y_U. \tag{7}$$

These equalities concern optimal objective values; the dual vectors $y_L, y_U \in \mathbb{R}^m$ have unrestricted-sign weights on equality constraints.

A separate dual-certificate network summarizes the LP using shared column encodings with mean and coordinatewise maximum pooling:

$$e_j = \phi_{\text{col}}([A_{:j} \,\|\, c_j]), \qquad h_{\text{LP}} = \left[\frac{1}{K}\sum_{j=1}^{K} e_j \,\|\, \max_{j=1,\dots,K} e_j\right]. \tag{8}$$

This summary is invariant to joint permutations of response-type columns but not to constraint-row ordering. Appendix F specifies the fixed serialization used in all experiments. The certificate network does not consume or modify PotencyGNN's point prediction.

Let $e_0 \in \mathbb{R}^m$ select the normalization row, so $A^\top e_0 = \mathbf{1}_K$ and $b^\top e_0 = 1$, and define $\alpha_L = \min_j c_j$ and $\alpha_U = \max_j c_j$. Since $\alpha_L \mathbf{1}_K \leq c \leq \alpha_U \mathbf{1}_K$, the anchors $\alpha_L e_0$ and $\alpha_U e_0$ are dual feasible. From $\xi = [h_{\text{LP}} \,\|\, b \,\|\, \alpha_L \,\|\, \alpha_U]$, the network predicts unrestricted residuals (r_L, r_U) and forms $\widetilde{y}_L = \alpha_L e_0 + r_L$ and $\widetilde{y}_U = \alpha_U e_0 + r_U$. These may violate dual feasibility, so we correct the largest componentwise violations through the normalization coordinate:

$$\begin{aligned} s_L &= \max\left\{0, \max_j\left[(A^\top \widetilde{y}_L)_j - c_j\right]\right\}, \quad y_L^{\text{rep}} = \widetilde{y}_L - s_L e_0, \\ s_U &= \max\left\{0, \max_j\left[c_j - (A^\top \widetilde{y}_U)_j\right]\right\}, \quad y_U^{\text{rep}} = \widetilde{y}_U + s_U e_0. \end{aligned} \tag{9}$$

In exact arithmetic, these shifts enforce $A^\top y_L^{\text{rep}} \leq c$ and $A^\top y_U^{\text{rep}} \geq c$ because the normalization coordinate shifts every constraint equally. These statements hold in exact arithmetic; the double-precision verifier outwardly bounds rounding error in every dual constraint and objective (Appendix F). Let $L_{\text{dual}} = b^\top y_L^{\text{rep}}$ and $U_{\text{dual}} = b^\top y_U^{\text{rep}}$. Weak duality gives $L_{\text{dual}} \leq L_{\text{full}} \leq U_{\text{full}} \leq U_{\text{dual}}$; intersecting with the exact base interval yields

$$[L_{\text{cert}}, U_{\text{cert}}] := [\max\{L_{\text{dual}}, L_{\text{base}}\}, \min\{U_{\text{dual}}, U_{\text{base}}\}] \supseteq \mathcal{I}_{\text{full}}(Q). \tag{10}$$

Hence

$$w_{\text{full}} \leq w_{\text{cert}} := U_{\text{cert}} - L_{\text{cert}} \leq w_{\text{base}}, \qquad 0 \leq \tau_{\text{cert}} := w_{\text{base}} - w_{\text{cert}} \leq \tau. \tag{11}$$

Under the stated model and support conditions, and in exact arithmetic, $\tau_{\text{cert}} > 0$ guarantees at least that much tightening, whereas $\tau_{\text{cert}} = 0$ is inconclusive. The certificate provides no statistical coverage, does not validate the assumed graph, and does not bound PotencyGNN's prediction error.

Without endpoint or dual supervision, the network minimizes the raw repaired width $U_{\text{dual}} - L_{\text{dual}}$ before base intersection (Appendix F, Equation 13). Learning controls tightness, while repair supplies exact-arithmetic validity. At inference, the certificate retains the two base-regime LP solves and replaces the two full-regime solves with repaired dual bounds. To account for rounding, we outwardly verify the repaired-dual interval and a base outer interval, then intersect them. Conditional on full-LP feasibility, the intersection contains the full identified set; subtracting its width from a verified base-width lower bound gives $\tau_{\text{cert}}^{\text{fp}}$, whose positive values certify tightening for the stored LPs. Appendix F provides the architecture, training details, soundness proof, and numerical verification procedure.

Table 1: Held-out-core prediction within the screened three-to-five-variable (3–5V) and exact-six-variable (6V) benchmarks. Neural entries are mean held-out $R^2\pm$ population standard deviation over five initializations, averaged across three core-disjoint splits; GBM reports a single fit per split with `random_state=0`. Size denotes trainable parameters or GBM leaves and may vary across splits and benchmarks.

Model	Size	Device	Benchmark	Split 7	Split 11	Split 13	Mean
PotencyGNN	67,841	CPU	3–5V	0.829 ± 0.002	0.822 ± 0.003	0.837 ± 0.002	0.829
			6V	$\mathbf{0.784 \pm 0.020}$	$\mathbf{0.786 \pm 0.008}$	$\mathbf{0.765 \pm 0.006}$	**0.779**
GRIT (tuned)	121,097–670,733	GPU	3–5V	$\mathbf{0.842 \pm 0.008}$	$\mathbf{0.831 \pm 0.019}$	$\mathbf{0.855 \pm 0.013}$	**0.843**
			6V	0.764 ± 0.004	0.741 ± 0.022	0.758 ± 0.009	0.754
PNA (tuned)	35,713–560,641	GPU	3–5V	0.755 ± 0.012	0.748 ± 0.016	0.749 ± 0.010	0.751
			6V	0.680 ± 0.023	0.654 ± 0.024	0.668 ± 0.008	0.667
GBM	5,487–9,021 leaves	CPU	3–5V	0.728	0.718	0.680	0.709
			6V	0.642	0.671	0.617	0.643
GIN	42,884	CPU	3–5V	0.700 ± 0.020	0.712 ± 0.014	0.698 ± 0.023	0.703
			6V	0.560 ± 0.025	0.629 ± 0.004	0.566 ± 0.032	0.585

5 Evaluation

5.1 Experimental setup

We use ADMGs with three to five observed binary variables, at most two observed parents per variable in C, and $K \leq 256$ joint response types; each instance comprises (G, X, Y), SCM-induced $P(V)$, and an exact potency label. The *causal core* is the subgraph induced by $\{X, Y\} \cup C \cup \mathrm{an}_G(X) \cup \mathrm{an}_G(Y) \cup \mathrm{pa}_G(C)$; remaining variables are *decorations*. A retained set is *parentally closed* if it contains every observed parent of each retained variable. Under parental closure and Assumption S, causal-core reduction preserves all four exact endpoints and hence potency; Appendix C gives the theorem and proof. All three-to-five-variable reductions satisfy this condition ($246/246$ shape-level and $25/25$ decorated core-first representatives). Appendix B gives the six-variable construction and checks. After canonicalizing causal cores over permutations of auxiliary-variable labels with X, Y fixed, a 32-SCM screen retains 440 causal cores with at least 16 finite labels and at least one potency above 10^{-4}. Screening draws are not reused; 60 fresh SCMs per retained core yield 26,400 labeled instances. Three core-disjoint splits (seeds $\{7, 11, 13\}$) assign 330 cores to training and 110 to evaluation, with no overlap. The exact-six-variable benchmark uses the same screen and separate model training. All synthetic instances satisfy Assumption S with $\mathcal{F}_{\mathrm{full}} \neq \emptyset$ by construction; all 19 empirical rows satisfy S and produce finite numerical base/full LP solutions (Appendix G).

Baselines comprise GBM(Friedman, 2001; Holzmüller et al., 2024), a fixed-order flat MLP, DeepSets(Zaheer et al., 2017; Wagstaff et al., 2022), GIN (Xu et al., 2019), PNA(Corso et al., 2020), and GRIT(Ma et al., 2023). Neural models share node and global features, ordered-query readout, objective, and benchmark instances, while GBM uses corresponding aggregated features. PotencyGNN and fixed comparator configurations are prespecified; tuned configurations are selected only on synthetic training-side validation cores and refitted on all training cores, with no fitting or selection on empirical data. We report held-out R^2 on synthetic benchmarks and per-query absolute error with dataset-macro MAE on empirical evaluations, averaging within datasets before weighting datasets equally. For published-network screening, we report ROC-AUC and AUPRC for $\tau > 10^{-6}$ and $\tau > 0.05$: the former excludes numerical-zero cases, whereas the latter requires a width reduction greater than 0.05. Neither threshold is used for model selection. Appendix D gives full metric definitions, hyperparameters, and split-, subset-, and initialization-level results.

5.2 Generalization to unseen causal cores

On the three-to-five-variable (3–5V) benchmark, PotencyGNN achieves mean held-out $R^2 = 0.829$, compared with 0.843 for tuned GRIT (Table 1). GRIT has higher R^2 on all three splits; bootstrap intervals for the GRIT–PotencyGNN R^2 difference exclude zero on splits 7 and 13, but not on split 11. PotencyGNN nevertheless exceeds PNA, GBM, GIN, and the flat MLP on every split. Tuning improves GBM's mean R^2 by at most 0.008, leaving it well below PotencyGNN. The gaps

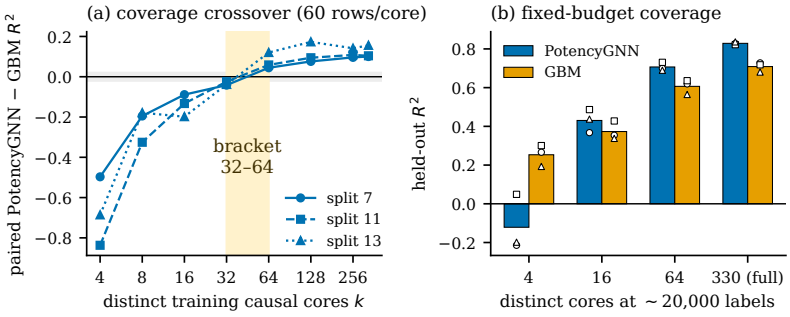

Figure 1: **Structural coverage and predictive performance.** (a) Mean paired held-out R^2 difference (PotencyGNN minus fixed GBM) with 60 labels per core, shown separately for three core-disjoint splits. The difference changes sign between 32 and 64 cores on every split. (b) Held-out R^2 at approximately 20,000 labels; bars show means across splits and open markers show individual splits.

to the flat MLP and untyped GIN are consistent with benefits from relational structure and edge orientation, while GRIT also uses typed pair information, so its lead cannot be attributed uniquely to attention, capacity, or device; Appendix D reports the full comparisons and ablations. Instances within a causal core share the same graph structure but differ in their mechanisms, so a graph-only predictor must assign the same value to all instances in that core. Even an oracle that predicts each held-out core's true mean potency achieves only $R^2 = 0.24$–0.27, compared with 0.829 for PotencyGNN, showing that the observational features capture variation among mechanisms sharing the same structure. On the separately generated exact-six benchmark, PotencyGNN leads with mean $R^2 = 0.779$ versus 0.754 for GRIT and exceeds GRIT, PNA, GBM, and GIN on all three splits; however, the bootstrap interval for the PotencyGNN–GRIT R^2 difference excludes zero only on split 11. Appendix B gives additional six-variable results. Because the 3–5V and 6V benchmarks are generated separately, these comparisons establish neither a graph-size trend nor unique superiority of PotencyGNN's architecture. Within the screened benchmarks, PotencyGNN generalizes to held-out causal cores and remains competitive with GRIT using observational features.

5.3 Structural Coverage and Sample Allocation

With 60 labels per core, increasing the number of training cores increases both structural coverage and sample size. In the coverage sweep comparing PotencyGNN with the fixed GBM, GBM has higher mean held-out R^2 through 32 cores on each split, whereas PotencyGNN leads from 64 cores onward (Figure 1(a)). The observed change in ordering therefore lies between 32 and 64 training cores in this benchmark.

To assess coverage at approximately matched label budgets, we compare 330 cores with 60 instances each (19,800 labels) against $k \in \{4, 16, 64\}$ cores with 19,968–20,000 freshly sampled instances. At four cores, mean held-out R^2 is -0.121 for PotencyGNN and 0.253 for GBM. Moving from four to 330 cores improves both models, with a larger PotencyGNN gain on each split (Figure 1(b)). In this benchmark, sampling more mechanisms from a few cores does not recover the performance obtained with broader structural coverage. Appendix D reports the per-split coverage differences, fixed-budget gains, and additional full-coverage results.

5.4 Architectural Ablations and Computational Efficiency

Removing message-passing adjacency while retaining the full parameter count lowers mean R^2 from 0.829 to 0.277, while symmetrizing the directed-neighbor aggregates lowers it by 0.065 despite retaining directed degree features (Table 4; Appendix D). These results support relational propagation and direction-specific aggregation. By contrast, zeroing ρ_i^{xy} changes mean R^2 by only -0.006 and zeroing $\mathbf{g}$ by $+0.004$, providing little evidence that either component is individually necessary on this benchmark; all headline results use the complete prespecified model. Appendix D gives the full

ablations and parameter-count caveats, including that equal parameter counts do not imply equal effective capacity.

From identical $(G, P(V), X, Y)$ inputs on one CPU thread, median batch-one latency is 0.824 ms for PotencyGNN feature construction and prediction versus 6.776 ms for cached-graph exact evaluation, an $8.22\times$ speedup (95% grouped-core bootstrap CI $[7.80, 8.77]$). Including graph-specific LP construction raises exact latency to 18.789 ms, yielding a $22.80\times$ cold-graph speedup (95% CI $[17.35, 24.63]$). At batch size 256, PotencyGNN requires 0.281 ms per query versus 7.346 ms for cached exact evaluation, giving $26.17\times$ throughput. Certificate inference takes 3.977 ms per query and retains the two exact base solves, so we report no exact-versus-certificate speedup. A 129.2 s fit breaks even after approximately $21{,}700$ batch-one evaluations when labels are available, or $47{,}800$ including the 155.2 s required to label the $19{,}800$ training instances; Appendix D gives component timings (Table 5).

5.5 Large-response-space stress test

Training uses $K \leq 256$. Across $2{,}242$ canonical instances spanning $K = 16$ to $262{,}144$, PotencyGNN yields positive zero-shot R^2 point estimates just beyond training: 0.580 at $K = 512$ and 0.668 at $K = 1024$, versus 0.761–0.870 within it. Its inference cost remains near 1 ms across the full range, while cached exact evaluation grows from 3.57 ms to $4{,}265.83$ ms. Farther beyond training, predictive quality becomes unstable: $R^2 = -0.036$ at $K = 2048$, recovers to 0.325 at $K = 4096$, and falls to -9.579 at $K = 262{,}144$, where MAE rises from 0.034 to 0.118 and rank agreement is lost. PotencyGNN exceeds the prespecified GBM comparator in R^2 at every tested K value, but its own absolute accuracy also deteriorates, so this relative advantage does not imply reliable far-out extrapolation. Overall, the surrogate retains positive R^2 through $K = 1024$ and near-constant inference cost across the tested range, while accuracy becomes unreliable farther beyond training. Appendix H gives the full per-K results, model identity, exposure status, and predefined sensitivity analyses.

5.6 Transfer to empirical datasets

We evaluate synthetic-trained models without empirical adaptation on five public datasets: Vitamin A (Sommer & Zeger, 1991; Balke & Pearl, 1997), Lipid Research Clinics (Ramsahai & Spirtes, 2012), Angrist–Evans fertility (Angrist & Evans, 1998), the National JTPA Study (Bloom et al., 1997; Abadie et al., 2002), and Sachs protein signaling (Sachs et al., 2005). Reference targets are exact potencies of the estimated $\widehat{P}(V)$ under the specified graphs, not ground-truth causal effects. We report seven primary queries across five datasets; outcome, coding, subgroup, and discretization variants appear in Appendix E.

Averaging all 15 PotencyGNN models (three core splits $\times$ five initializations), with no empirical model selection, yields dataset-macro MAE 0.0125, versus 0.0388 for the three split-matched GBMs. PotencyGNN has lower error on Vitamin A, Lipid, and Sachs, but higher error on Fertility and JTPA. Across the nine baseline configurations, dataset-macro MAE ranges from 0.0205 for GRIT to 0.0432 for tuned flat MLP; GRIT is lower than PotencyGNN on the split-7 held-out-core sub-ensemble, and several models are lower on individual datasets (Appendix E, Table 9).

The suite has limited structural diversity: the four positive-query datasets share one binary instrumental-variable core, which is held out from training in split 7, while the Sachs cores are absent from synthetic training. The dedicated split-7 comparison on the four primary IV datasets is reported in Appendix E, Table 8. A closed-form compliance margin exactly reproduces the positive-query potencies, whereas a constant-zero predictor is exact on Sachs. These experiments therefore test transfer and calibration rather than the necessity of amortized prediction; Appendix E gives the full provenance and split-dependent sensitivities.

We also evaluate 400 deduplicated, numerically feasible queries from 20 published bnlearn networks (Scutari, 2010) under posited $X \leftrightarrow Y$ confounding: 144 have potency above 10^{-6} and 256 are at or below it. Detecting potency above 10^{-6} remains difficult, with ROC-AUC 0.682 and 0.582 under the uniform and calibrated parameter distributions, despite stronger regression within the positive stratum (Appendix E, Table 10). Appendix E also relates this setting to the experiment-indexed pruning criteria of Maringgele & Etesami (2026).

5.7 CERTIFIED BOUNDS AND SCREENING UTILITY

Across three core-disjoint splits, nine certificate models each evaluate 8,800 held-out instances. The verifier outwardly bounds roundoff and exactly checks base primal witnesses, but does not certify full-LP feasibility. Across 79,200 evaluations, every reported interval contains the numerical reference interval, reported potency never exceeds numerical reference potency, and no numerical failure occurs. Intersecting the certificate with the exact base interval makes the certified interval tighter. Before intersection, the median excess width is computed within each model and then averaged across the nine models, giving 0.106 ± 0.011; after intersection, it decreases to 0.092 ± 0.007. Here, excess width is certified width minus exact width, and $\pm$ denotes the population standard deviation across models. The two evaluations agree on the post-intersection value to three decimals. For screening, $43.4\% \pm 4.4\%$ of held-out queries exceed the prespecified reporting threshold $\tau_{\mathrm{cert}}^{\mathrm{fp}} > 10^{-5}$, versus none for the normalization-row anchor. We use 10^{-5} only to avoid reporting numerically negligible positive values; any verified $\tau_{\mathrm{cert}}^{\mathrm{fp}} > 0$ already certifies positive width reduction, while nonpositive values are inconclusive. Recall@10% is 0.487 ± 0.028, compared with 0.10 under random ranking, indicating useful but incomplete screening. In exact arithmetic, $\tau_{\mathrm{cert}} \geq 0$ by construction. Tightness and ranking are established only on this synthetic core-disjoint benchmark; Appendix F gives the full numerical verification and regression audit.

6 CONCLUSION

We introduced *PotencyGNN*, a query-conditioned relational surrogate for epistemic potency under the stated model and support conditions, together with a separate dual-certificate network that provides certified potency lower bounds. Across core-disjoint benchmarks, PotencyGNN achieves held-out $R^2 = 0.779$–0.829, remains competitive with the evaluated graph transformer, and outperforms the other evaluated baselines. Under matched CPU inputs, its inference is $8.22\times$ faster than cached exact evaluation for individual queries and achieves $26.17\times$ higher batched throughput. At matched label budgets, broader causal-core coverage improves performance more than sampling additional mechanisms from a few cores. The stress test yields positive R^2 just beyond training, but unstable accuracy on larger response spaces. Future work includes larger and multivalued graphs, multiple confounded components, more structurally diverse empirical settings, amortized base bounds, and tighter certificates.

AI USE STATEMENT

In this work, generative AI tools were used for language editing, including grammar, spelling, and word choice, as well as for coding assistance. All scientific content, experiments, analyses, results, and conclusions were developed and verified by the authors.

ETHICS STATEMENT

This work uses synthetic benchmarks generated from binary acyclic directed mixed graphs together with publicly available networks and previously collected datasets. We conduct no new human-subject data collection and use no personally identifiable information.

REPRODUCIBILITY STATEMENT

Anonymized code is provided as supplementary material. The paper and appendix specify the causal assumptions, bounding programs and proofs, synthetic benchmark generation, core-disjoint splits, model configurations, and evaluation procedures. The empirical analyses use publicly available datasets and Bayesian networks cited in the paper.

ACKNOWLEDGMENTS

Use unnumbered third level headings for the acknowledgments. All acknowledgments, including those to funding agencies, go at the end of the paper.

REFERENCES

Alberto Abadie, Joshua Angrist, and Guido Imbens. Instrumental variables estimates of the effect of subsidized training on the quantiles of trainee earnings. *Econometrica*, 70(1):91–117, 2002.

Sina Akbari, Jalal Etesami, and Negar Kiyavash. Optimal experiment design for causal effect identification. *Journal of Machine Learning Research*, 26(28):1–56, 2025.

Joshua D Angrist and William N Evans. Children and their parents' labor supply: Evidence from exogenous variation in family size. *American Economic Review*, 88(3):450–477, June 1998. URL https://ideas.repec.org/a/aea/aecrev/v88y1998i3p450-77.html.

João Pedro Arroyo, Denis Deratani Mauá, João Gabriel Barreto Rodrigues, Daniel A. E. Lawand, Junkyu Lee, Radu Marinescu, Alex Gray, Eduardo Rocha Laurentino, and Fabio Gagliardi Cozman. Multilinear and linear programs for partially identifiable queries in quasi-markovian structural causal models. In *Conference on Uncertainty in Artificial Intelligence - UAI*. AUAI Press, 2025.

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

Vahid Balazadeh Meresht, Hamidreza Kamkari, Valentin Thomas, Junwei Ma, Bingru Li, Jesse Cresswell, and Rahul Krishnan. Causalpfn: Amortized causal effect estimation via in-context learning. *Advances in Neural Information Processing Systems*, 38:154945–154984, 2026.

Alexander Balke and Judea Pearl. Counterfactual probabilities: Computational methods, bounds and applications. In *Uncertainty in artificial intelligence*, pp. 46–54. Elsevier, 1994.

Alexander Balke and Judea Pearl. Bounds on treatment effects from studies with imperfect compliance. *Journal of the American statistical Association*, 92(439):1171–1176, 1997.

Peter W Battaglia, Jessica B Hamrick, Victor Bapst, Alvaro Sanchez-Gonzalez, Vinicius Zambaldi, Mateusz Malinowski, Andrea Tacchetti, David Raposo, Adam Santoro, Ryan Faulkner, et al. Relational inductive biases, deep learning, and graph networks. *arXiv preprint arXiv:1806.01261*, 2(3):5, 2018.

Alexis Bellot and Anish Dhir. Foundation models for partial causal identification. In *2nd ICML Workshop on Foundation Models for Structured Data*, 2026. URL https://openreview.net/forum?id=jCbehzZBsk.

Howard S Bloom, Larry L Orr, Stephen H Bell, George Cave, Fred Doolittle, Winston Lin, and Johannes M Bos. The benefits and costs of jtpa title ii-a programs: Key findings from the national job training partnership act study. *Journal of human resources*, pp. 549–576, 1997.

Gabriele Corso, Luca Cavalleri, Dominique Beaini, Pietro Liò, and Petar Veličković. Principal neighbourhood aggregation for graph nets. *Advances in neural information processing systems*, 33:13260–13271, 2020.

Michael Dinitz, Sungjin Im, Thomas Lavastida, Benjamin Moseley, and Sergei Vassilvitskii. Faster matchings via learned duals. *Advances in neural information processing systems*, 34:10393–10406, 2021.

Priya L. Donti, David Rolnick, and J Zico Kolter. DC3: A learning method for optimization with hard constraints. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=V1ZHVxJ6dSS.

Jerome H Friedman. Greedy function approximation: a gradient boosting machine. *Annals of statistics*, pp. 1189–1232, 2001.

David Holzmüller, Léo Grinsztajn, and Ingo Steinwart. Better by default: Strong pre-tuned mlps and boosted trees on tabular data. In *Neural Information Processing Systems*, 2024.

Lars Lorch, Scott Sussex, Jonas Rothfuss, Andreas Krause, and Bernhard Schölkopf. Amortized inference for causal structure learning. *Advances in Neural Information Processing Systems*, 35: 13104–13118, 2022.

Liheng Ma, Chen Lin, Derek Lim, Adriana Romero-Soriano, Puneet K Dokania, Mark Coates, Philip Torr, and Ser-Nam Lim. Graph inductive biases in transformers without message passing. In *International Conference on Machine Learning*, pp. 23321–23337. PMLR, 2023.

Charles F Manski. Nonparametric bounds on treatment effects. *The American Economic Review*, 80(2):319–323, 1990.

Tobias Maringgele and Jalal Etesami. Optimal experiments for partial causal effect identification. *arXiv preprint arXiv:2605.06993*, 2026.

Vahid Balazadeh Meresht, Hamidreza Kamkari, Mr. Shailendra Barath, Ricardo Silva, and Rahul G. Krishnan. Iv-icl: Bounding causal effects with instrumental variables via in-context learning. *ArXiv*, abs/2605.12924, 2026. URL https://api.semanticscholar.org/CorpusID:288255205.

Roland R Ramsahai and Peter Spirtes. Causal bounds and observable constraints for non-deterministic models. *Journal of Machine Learning Research*, 13(3), 2012.

Thomas Richardson. Markov properties for acyclic directed mixed graphs. *Scandinavian Journal of Statistics*, 30(1):145–157, 2003.

Emanuele Rossi, Bertrand Charpentier, Francesco Di Giovanni, Fabrizio Frasca, Stephan Günnemann, and Michael M Bronstein. Edge directionality improves learning on heterophilic graphs. In *Learning on graphs conference*, pp. 25–1. PMLR, 2024.

Siegfried M Rump. Accurate solution of dense linear systems, part ii: Algorithms using directed rounding. *Journal of computational and applied mathematics*, 242:185–212, 2013.

Karen Sachs, Omar Perez, Dana Pe'er, Douglas A Lauffenburger, and Garry P Nolan. Causal protein-signaling networks derived from multiparameter single-cell data. *Science*, 308(5721): 523–529, 2005.

Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne Van Den Berg, Ivan Titov, and Max Welling. Modeling relational data with graph convolutional networks. In *European semantic web conference*, pp. 593–607. Springer, 2018.

Marco Scutari. Learning bayesian networks with the bnlearn r package. *Journal of statistical software*, 35(1):1–22, 2010.

Madhumitha Shridharan and Garud Iyengar. Scalable computation of causal bounds. *Journal of Machine Learning Research*, 24(237):1–35, 2023a.

Madhumitha Shridharan and Garud Iyengar. Causal bounds in quasi-markovian graphs. In *International Conference on Machine Learning*, pp. 31675–31692. PMLR, 2023b.

Alfred Sommer and Scott L Zeger. On estimating efficacy from clinical trials. *Statistics in medicine*, 10(1):45–52, 1991.

Jiyuan Tan, Jose Blanchet, and Vasilis Syrgkanis. Consistency of neural causal partial identification. *Advances in Neural Information Processing Systems*, 37:68956–68999, 2024.

Mathieu Tanneau and Pascal Van Hentenryck. Dual lagrangian learning for conic optimization. *Advances in Neural Information Processing Systems*, 37:55538–55561, 2024.

Edward Wagstaff, Fabian B Fuchs, Martin Engelcke, Michael A Osborne, and Ingmar Posner. Universal approximation of functions on sets. *Journal of Machine Learning Research*, 23(151):1–56, 2022.

Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=ryGs6iA5Km.

Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, Barnabas Poczos, Russ R Salakhutdinov, and Alexander Smola. Deep sets. *Advances in neural information processing systems*, 30, 2017.

APPENDIX

A WORKED EXAMPLE: RESPONSE TYPES, SHARP BOUNDS, AND POTENCY

This section gives a complete binary example of the construction in Section 3, from latent response types to observed probabilities, sharp causal bounds, and realized observational epistemic potency. The main distinction is between the response-type distribution q, which is unobserved, and the vectors b_{base} and b_{full}, which are determined by the observational distribution.

Graph and SCM conventions. The directed part of G is acyclic, so the structural assignments induce a well-defined observational distribution $P(V)$. Dependence among the exogenous variables can induce latent confounding. A bidirected edge represents latent dependence and does not imply causal influence in either direction. The confounded clique C consists of observed variables: every pair of nodes in C is connected by a bidirected edge, while no bidirected edges occur outside C. Under the restricted SCM class of Section 3, the exogenous variables outside C are mutually independent and independent of U_C, and their corresponding conditional distributions are fixed from $P(V)$.

Compatible models and endpoints. The observational distribution is generated by an unknown SCM $\mathcal{M}^\star$ in the response-function class of Section 3. The graph and retained observational information generally do not uniquely determine the latent response-type distribution. For each regime $s \in \{\text{base}, \text{full}\}$, $\mathfrak{M}_s$ contains all models in this class that satisfy the corresponding observational constraints, and $[L_s, U_s]$ is the range of query values attainable over $\mathfrak{M}_s$. The lower and upper endpoints may therefore be attained by different compatible SCMs. This reflects partial identification: multiple SCMs can agree on the same retained observational information while implying different values of the causal query. These SCMs differ only in latent mechanisms that the available observational information cannot distinguish.

Response functions and LP constraints. A deterministic response function specifies a variable's value for every possible configuration of its observed parents. For example, a binary variable with one binary parent has four possible functions, $(0,0)$, $(0,1)$, $(1,0)$, and $(1,1)$, where the two entries give its value when the parent equals 0 and 1, respectively. Each such function is a *response type*. A joint response type selects one response type for each variable in the clique C.

Let q_j denote the probability assigned to joint response type j, and let q collect these probabilities. The response-type distribution q is not observed directly; in the linear programs, it is the decision variable over which we optimize. For regime $s \in \{\text{base}, \text{full}\}$, the feasible response-type distributions are

$$\mathcal{F}_s = \left\{ q \geq 0 : \mathbf{1}^\top q = 1,\ A_s q = b_s \right\}.$$

The constraints $q \geq 0$ and $\mathbf{1}^\top q = 1$ ensure that q is a valid probability distribution, while $A_s q = b_s$ requires it to reproduce the observational probabilities retained under regime s. Here, b_s contains those probabilities, and each column of A_s records how one joint response type contributes to them. The coefficient c_j gives the causal-query value associated with joint type j, so the query value implied by a feasible response-type distribution is $c^\top q$.

Consider the binary ADMG

$$Z \to X \to Y, \qquad X \leftrightarrow Y,$$

where Z is an observed auxiliary variable, X is the treatment, and Y is the outcome. Let

$$P(Z=0) = P(Z=1) = 1/2,$$

let the clique be $C = \{X, Y\}$, and consider the query

$$Q = P(Y = 1 \mid \text{do}(X = 1)).$$

For $z \in \{0,1\}$, $X(z)$ denotes the value that X would take if Z were set to z. Similarly, for $x \in \{0,1\}$, $Y(x)$ denotes the value that Y would take if X were set to x. Thus, $Y(0)$ and $Y(1)$ are the potential outcomes under $X = 0$ and $X = 1$, respectively. For the realized value of X, the observed outcome is $Y = Y(X)$; the other potential outcome remains counterfactual. The causal query can therefore be written equivalently as

$$Q = P(Y(1) = 1).$$

In this example, because X has the binary parent Z, its response type is

$$r_X = (X(0), X(1)).$$

For example, $r_X = (0,1)$ means that $X = 0$ when $Z = 0$ and $X = 1$ when $Z = 1$. Similarly, because Y has the binary parent X, its response type is

$$r_Y = (Y(0), Y(1)).$$

Thus, $r_Y = (1,0)$ means that $Y = 1$ under $X = 0$ and $Y = 0$ under $X = 1$. For r_X, the two positions correspond to $Z = 0$ and $Z = 1$; for r_Y, they correspond to $X = 0$ and $X = 1$.

Each variable therefore has the four possible response types $(0,0)$, $(0,1)$, $(1,0)$, and $(1,1)$. A joint response type is a pair (r_X, r_Y), selecting one response type for X and one for Y. There are therefore

$$4 \times 4 = 16$$

possible joint response types.

For illustration, suppose the generating response-type distribution has the following nonzero probabilities; the remaining ten joint types have probability zero.

r_X	r_Y	$q(r_X, r_Y)$
$(0,0)$	$(1,0)$	0.080
$(0,1)$	$(0,0)$	0.330
$(0,1)$	$(0,1)$	0.160
$(0,1)$	$(1,1)$	0.340
$(1,1)$	$(0,1)$	0.085
$(1,1)$	$(1,1)$	0.005

These probabilities sum to one. The query asks for the probability that $Y(1) = 1$, so we sum the probabilities of the joint types whose second entry in r_Y equals 1:

$$Q = 0.160 + 0.340 + 0.085 + 0.005 = 0.590.$$

The same response-type distribution also determines the observational distribution. It induces

	$(X,Y) = (0,0)$	$(0,1)$	$(1,0)$	$(1,1)$
$Z = 0$	0.245	0.210	0	0.045
$Z = 1$	0	0.040	0.165	0.295

For example, when $Z = 0$, the joint types with probabilities 0.330 and 0.160 both produce $X = 0$ and $Y = 0$. Since $P(Z = 0) = 1/2$,

$$P(Z = 0, X = 0, Y = 0) = \frac{1}{2}(0.330 + 0.160) = 0.245.$$

Thus, in lexicographic order,

$$b_{\text{full}} = (0.245, 0.210, 0, 0.045, 0, 0.040, 0.165, 0.295)^\top.$$

The full regime therefore retains all eight cells of $P(Z, X, Y)$.

The base regime imposes the $P(X, Y)$ cell constraints. Marginalizing over Z gives

$$b_{\text{base}} = (0.245, 0.250, 0.165, 0.340)^\top.$$

For example,

$$P(X=0,Y=1) = P(Z=0,X=0,Y=1)+P(Z=1,X=0,Y=1) = 0.210+0.040 = 0.250.$$

Thus, b_{base} contains the four cells of $P(X,Y)$, whereas b_{full} retains the additional observational information involving Z.

The constraint matrices map the latent response-type probabilities to these observational probabilities. For a joint type (r_X, r_Y),

$$(A_{\text{full}})_{(z,x,y),(r_X,r_Y)} = P(Z=z)\,\mathbf{1}\{r_X(z)=x\}\,\mathbf{1}\{r_Y(x)=y\},$$

and

$$(A_{\text{base}})_{(x,y),(r_X,r_Y)} = \sum_z P(Z=z)\,\mathbf{1}\{r_X(z)=x\}\,\mathbf{1}\{r_Y(x)=y\}.$$

Here, $\mathbf{1}\{\cdot\}$ denotes the indicator function. Intuitively, each column records which observed event is produced by one particular joint response type. The full matrix separates values of Z, whereas the base matrix sums over Z to impose only the $P(X,Y)$ cell constraints.

Because $Z \notin C$ and has no observed parents, its fixed outside-C conditional distribution is simply its marginal distribution $P(Z)$. Both regimes use the same $P(Z)$, consistent with the response-function class of Section 3. They differ only in the observational probabilities that q must reproduce: the base regime requires reproduction of $P(X,Y)$, whereas the full regime requires reproduction of $P(Z,X,Y)$. The generating response-type distribution displayed above is one feasible q, but it is not supplied to the bounding programs. Instead, each program optimizes over all $q \in \mathcal{F}_s$ satisfying the corresponding observational constraints. Several feasible response-type distributions can therefore reproduce the same retained observational probabilities while implying different values of the causal query.

As a diagnostic, we also checked whether the fixed value of $P(Z)$ affects the base interval in this example. Varying $P(Z=1)$ over $[0.2, 0.8]$ while holding b_{base} fixed leaves both base endpoints unchanged at double precision. The same endpoints are obtained in a separate variant that also assigns response types to Z, so that $P(Z)$ is not fixed. These checks show that the base interval is unchanged in this example, but they do not establish such invariance in general.

For the present query, the coefficient associated with joint type (r_X, r_Y) is

$$c_{(r_X,r_Y)} = r_Y(1),$$

because the intervention sets $X=1$ and the second entry of r_Y determines the resulting value of Y. Hence $c^\top q$ is the probability that $Y=1$ after setting $X=1$.

Solving the lower and upper linear programs under the base constraints gives

$$[L_{\text{base}}, U_{\text{base}}] = [0.340, 0.835], \qquad w_{\text{base}} = 0.495.$$

Using all cells of $P(Z,X,Y)$ gives

$$[L_{\text{full}}, U_{\text{full}}] = [0.590, 0.670], \qquad w_{\text{full}} = 0.080.$$

The generating response-type distribution displayed above happens to attain the full lower endpoint, $L_{\text{full}} = 0.590$. However, other feasible response-type distributions reproduce exactly the same full observational distribution while yielding query values as large as 0.670. Thus, even after using the full observational distribution, the latent response-type distribution is not uniquely determined and the causal query is not point identified in this example.

The epistemic potency is therefore

$$\tau(G,P,Q) = w_{\text{base}} - w_{\text{full}} = 0.495 - 0.080 = 0.415.$$

Hence, the additional observational information involving Z reduces the sharp identified-set width from 0.495 to 0.080, a reduction of 0.415. The example also makes clear why q and b_s play different roles: q assigns probabilities to unobserved joint response types, whereas b_s contains observational probabilities obtained by aggregating those latent response-type probabilities.

A.1 EVIDENCE-FEATURE DEFINITIONS

Section 4 introduces the query-conditioned evidence representation. Here we give the exact components and ordering of the 12-dimensional node evidence profile $\mathbf{x}_i$ and the 8-dimensional instance-level context vector $\mathbf{g}$. All probability-based features are computed from the observational distribution supplied to the model: synthetic benchmarks use the exact population distribution $P(V)$, whereas empirical evaluations use the corresponding empirical distribution $\hat{P}(V)$. Table 2 gives the complete specification.

Table 2: Exact specification of the node evidence profile $\mathbf{x}_i$ and instance-level context vector $\mathbf{g}$. Indices give the position within the corresponding vector.

Scope	Index	Definition
Node profile	0	$r_i^X = \mathbf{1}\{V_i = X\}$
Node profile	1	$r_i^Y = \mathbf{1}\{V_i = Y\}$
Node profile	2	$r_i^C = \mathbf{1}\{V_i \in C\}$
Node profile	3	$\mu_i = P(V_i = 1)$
Node profile	4	$\Delta_i^X = P(X = 1 \mid V_i = 1) - P(X = 1 \mid V_i = 0)$; set to zero when $V_i = X$
Node profile	5	$\Delta_i^Y = P(Y = 1 \mid V_i = 1) - P(Y = 1 \mid V_i = 0)$; set to zero when $V_i = Y$
Node profile	6–9	$\rho_i^{xy} = P(V_i = 1 \mid X = x, Y = y)$, ordered as $(x, y) = (0,0), (0,1), (1,0), (1,1)$
Node profile	10	d_i^{in}: directed in-degree of V_i
Node profile	11	d_i^{out}: directed out-degree of V_i
Global context	0–3	$\pi_{xy} = P(X = x, Y = y)$, ordered as $(x, y) = (0,0), (0,1), (1,0), (1,1)$
Global context	4	n: number of observed variables
Global context	5	$\lvert E_{\rightarrow}\rvert$: number of directed edges
Global context	6	$\lvert C\rvert$: number of variables in the clique C
Global context	7	$\log_2 K$, where $K = \prod_{V_i \in C} 2^{2^{k_i}}$ is the number of joint response types and $k_i = \lvert \text{pa}_G(V_i)\rvert$

Undefined conditionals. Some conditional features are undefined when their conditioning event has zero or negligible probability. For the query-cell prevalence $\rho_i^{xy} = P(V_i = 1 \mid X = x, Y = y)$, if $\pi_{xy} = P(X = x, Y = y) < 10^{-9}$, we assign the fixed placeholder value 0.5. This value is only a numerical convention and is not an estimate from the corresponding cell.

The same rule is applied to the two conditional terms used to construct Δ_i^T, for $T \in \{X, Y\}$. Specifically, if $P(V_i = v) < 10^{-9}$ for $v \in \{0, 1\}$, the corresponding term $P(T = 1 \mid V_i = v)$ is assigned 0.5 before the difference is formed. Self-associations are defined separately as $\Delta_X^X = \Delta_Y^Y = 0$.

No additional feature indicates whether a value of 0.5 is a placeholder or a genuine conditional probability. This convention applies only to PotencyGNN's input features. It is distinct from the treatment of undefined outside-C conditional distributions in the exact bounding programs, which is described in Appendix G.

B SIX-VARIABLE BENCHMARK EXTENSION

Construction. To evaluate PotencyGNN on larger graphs, we construct a separate collection starting from ADMGs with six observed binary variables. The generator follows the same structural restrictions as the three-to-five-variable benchmark: X and Y belong to the single bidirected clique C; each additional variable is included in C independently with probability 0.35; every variable in C has at most two observed parents; and the number of joint response types,

$$K = \prod_{V_i \in C} 2^{2^{k_i}}, \qquad k_i = |\text{pa}_G(V_i)|,$$

is restricted to $K \leq 256$. Thus, K is also the dimension of the LP decision vector q. This restriction keeps the response-type dimension within the range used by the primary synthetic benchmark and makes exact four-LP labeling computationally manageable. The initial number of directed edges is sampled uniformly from $\{5, \ldots, 12\}$. If the sampled graph contains no directed path from X to Y, we add the edge $X \to Y$.

Among 40,000 generation attempts, 9,615 yield distinct admissible graph topologies. We reduce each graph to its causal core and remove duplicate cores before screening. Each distinct core is screened using 32 independently sampled SCMs, which vary the numerical mechanisms while keeping the graph structure fixed. A core is retained if at least 16 of the 32 SCMs produce finite potency labels and at least one potency exceeds 10^{-4}. The first condition excludes cores for which exact labeling is frequently nonfinite under the screening draws, while the second removes cores whose screening labels are uniformly negligible. The value 10^{-4} is used only as a core-level screening threshold, not as a definition of positive potency.

For each retained core, we then sample 60 new SCMs to form the final labeled instances; the screening SCMs are not reused. Once a core passes the screen, every finite final label is retained, including zero and values below 10^{-4}. For every instance, PotencyGNN and the GBM receive features constructed from the same observational distribution, so their evaluations use paired inputs.

Core-size accounting. Causal-core reduction can remove decorations, so a graph initially generated with six variables may produce a smaller retained core. After reduction, core-level deduplication, and screening, the collection contains 959 representatives: 345 with six variables, 550 with five, 62 with four, and 2 with three.

A retained node set is *parentally closed* when every retained variable keeps all of its observed parents; equivalently, no removed variable has a directed child that remains in the core. Under this condition and Assumption S, the reduction theorem of Appendix C preserves all four exact endpoints and hence potency.

Among the 9,615 admissible graph topologies, 180 are not parentally closed. For 118 of these topologies, we compare the full graph and reduced core on three independently sampled validation SCMs. The corresponding endpoint calculations are finite, and every absolute full-versus-reduced endpoint difference is below the prespecified numerical-equivalence tolerance of 10^{-9}. This tolerance is used to distinguish numerical differences from changes in the computed endpoints. We therefore accept these reductions based on this numerical check.

For the remaining 62 topologies, at least one full-versus-reduced comparison is nonfinite, so equivalence cannot be verified and we retain the full graph. After deduplication and screening, only one such unreduced graph remains in the collection.

The thresholds 10^{-9} and 10^{-4} serve different purposes. The 10^{-9} tolerance is used only to assess numerical equivalence between the full and reduced calculations. The 10^{-4} threshold is used only during core-level screening to exclude cores whose screening draws show no non-negligible potency. Neither threshold modifies the final potency labels. In particular, after a core is retained, every finite label from its 60 final SCMs remains in the collection without thresholding. This separation ensures that the screening threshold does not truncate the final labels within retained cores.

The resulting collection is therefore a mixed-size collection derived from graphs initially generated with six variables, rather than a six-variable-only evaluation.

After deduplication and screening, exactly one accepted non-parentally-closed reduction remains in the 959-representative collection, in addition to the single unreduced graph described above. Neither appears in the exact six-variable evaluation subset. That subset contains 344 causal cores with exactly six variables, so all reductions used in the reported six-variable evaluation satisfy the conditions of the theorem in Appendix C.

For completeness, we directly checked the single accepted non-parentally-closed reduction on all 60 instances generated with the benchmark label seed. For each instance, we evaluate the original graph and its reduced core using the same sampled mechanism. Every corresponding full-versus-reduced endpoint difference remains below 10^{-9} and at the scale of double-precision rounding. We use this check only to justify retaining this benchmark representative and make no general invariance claim for non-parentally-closed reductions.

Evaluation restricted to six-variable causal cores. For this analysis, the inclusion rule and evaluation design were specified before model training and evaluation on the subset. We retain the 344 parentally closed causal cores containing exactly six variables; the exceptional non-parentally-closed reduction and the single unreduced representative described above do not enter this subset. These cores contain 20,640 previously generated, exactly labeled SCM instances, with 60 instances per core. No new SCMs are sampled and no labels are recomputed.

For each split seed in $\{7, 11, 13\}$, the 344 causal cores are partitioned into 258 training cores and 86 held-out cores, yielding 15,480 training instances and 5,160 held-out instances. No causal core is shared between training and evaluation. The held-out cores are therefore the units of structural generalization, while the 60 instances within each core vary the numerical mechanisms on the same graph structure.

PotencyGNN is trained independently on each split using five initializations. The paired GBM is trained and evaluated on exactly the same training and held-out instances. This experiment therefore measures generalization to unseen causal cores at a fixed graph size. It does not measure zero-shot transfer from models trained on the three-to-five-variable benchmark.

Table 3: Core-disjoint evaluation on the exact six-variable subset. PotencyGNN reports mean held-out $R^2\pm$ population standard deviation over five initializations; GBM uses one fit per split (`random_state=0`). The final row averages unrounded split values.

Split seed	PotencyGNN	GBM
7	0.784 ± 0.020	0.642
11	0.786 ± 0.008	0.671
13	0.765 ± 0.006	0.617
Mean	0.779	0.643

PotencyGNN has higher held-out R^2 than the paired GBM on all three core-disjoint splits, with mean $R^2 = 0.779$ versus 0.643 (Table 3). These results demonstrate generalization to held-out causal cores within the screened exact-six benchmark. Because the two benchmarks were generated and screened separately, with different topology distributions, core counts, and training-set sizes, their results do not identify a graph-size effect.

C CAUSAL-CORE REDUCTION

Validation procedure. Let $S \subseteq V$ denote the set of variables retained in the causal core defined in Section 5.1, and let $D = V \setminus S$ denote the removed variables, which we call decorations. We write $G[S]$ for the subgraph of G induced by the variables in S: it contains the variables in S and all directed and bidirected edges of G whose endpoints both lie in S.

For any joint assignment v of all variables in V, let v_S denote the values of the variables in S and v_D the values of the variables in D. For each sampled SCM, we first compute the four exact endpoints and potency using the full graph G and its induced observational distribution $P(V)$. We then reduce the graph to $G[S]$ and obtain the corresponding observational distribution on the retained variables by marginalizing out the decorations:

$$P_{\text{core}}(v_S) = \sum_{v_D} P(v_S, v_D).$$

Thus, P_{core} is not generated from a new SCM; it is obtained directly from the same observational distribution $P(V)$ used for the full graph. We then recompute the four endpoints and potency on $(G[S], P_{\text{core}})$ for the same ordered treatment–outcome query. The full and reduced calculations therefore differ only in whether the decorations are retained.

Let L_s^{core} and U_s^{core} denote the lower and upper endpoints obtained from $(G[S], P_{\text{core}})$ for $s \in \{\text{base}, \text{full}\}$. We measure the largest endpoint difference by

$$\delta_{\text{end}} = \max_{s \in \{\text{base,full}\}} \left\{ |L_s - L_s^{\text{core}}|, |U_s - U_s^{\text{core}}| \right\},$$

and the corresponding potency difference by

$$\delta_{\mathrm{pot}} = |\tau(G, P, Q) - \tau(G[S], P_{\mathrm{core}}, Q)| .$$

We report both quantities, but use the endpoint criterion $\delta_{\mathrm{end}} < 10^{-9}$ to treat a reduction as numerically equivalent. This tolerance is used only to distinguish finite-precision numerical differences from changes in the computed endpoints.

Parental closure. We call the retained node set S *parentally closed* when every observed parent of every retained variable is also retained:

$$\mathrm{pa}_G(V_i) \subseteq S \qquad \text{for every } V_i \in S.$$

Equivalently, no removed decoration in $D = V \setminus S$ has a directed edge into a variable that remains in S. For example, if $Z \to X$ and $X \in S$, then parental closure requires $Z \in S$ as well.

This condition ensures that reducing G to $G[S]$ does not remove any observed parent of a retained variable. Each retained variable therefore keeps the same parent set and the same response-function domain as in the full graph. This is important because the exact bounding programs are defined over response types whose domains depend on the observed parent sets.

If parental closure fails, a retained variable may lose an observed parent. Its response-function domain can then change, and the full and reduced bounding programs need not have the same response-type space or feasible set. The reduction may still preserve the exact bounds for a particular instance, but this is no longer guaranteed by the theorem below. We therefore use parental closure as a sufficient condition for exact causal-core reduction, not as a necessary one.

For each regime $s \in \{\mathrm{base}, \mathrm{full}\}$, let $\mathcal{F}_s$ denote the set of feasible response-type distributions in the corresponding bounding program, and let c denote the causal-query objective vector. The lower and upper endpoints for regime s are obtained by minimizing and maximizing $c^\top q$ over $q \in \mathcal{F}_s$.

Reduction theorem. Let G be an ADMG over binary V whose bidirected edges form a single clique $C \supseteq X, Y$, with $P_U = P_{U_C} \prod_{V_i \notin C} P_{U_i}$ and outside-C conditional distributions fixed by $P = P(V)$; let $Q = P(Y = 1 \mid \mathrm{do}(X = 1))$. For retained $S \subseteq V$, let $G[S]$ be the induced subgraph and P_{core} the marginal of P on S.

Suppose that S satisfies the following conditions: (S1) $C \cup \{X, Y\} \subseteq S$; (S2) S is parentally closed; (S3)

$$P\big(V_{\mathrm{pa}_G(V_i)} = a\big) > 0$$

for every $V_i \in S \setminus C$ and every assignment a to its observed parents; and (S4) $\mathcal{F}_{\mathrm{full}}(G, P) \neq \emptyset$.

Condition (S1) retains the bidirected clique and the query variables; (S2) ensures that every retained variable keeps all of its observed parents; (S3) ensures that the required outside-C conditional distributions of the retained variables are well defined; and (S4) ensures that the full-regime bounding program is feasible.

Then the bounding programs on (G, P) and $(G[S], P_{\mathrm{core}})$ have identical values of L_{base}, U_{base}, L_{full}, and U_{full}. Consequently,

$$\tau(G, P, Q) = \tau(G[S], P_{\mathrm{core}}, Q) .$$

Conditions (S1)–(S3) are sufficient to make the two base feasible sets identical; whenever these sets are nonempty, their lower and upper endpoints are therefore identical. Condition (S4) is required for the converse argument establishing equality of the full feasible sets. Parental closure is sufficient for the theorem but is not claimed to be necessary.

Proof. We first show that the full and reduced programs use the same response-type space. By (S2), every retained variable has the same observed parents before and after reduction:

$$\mathrm{pa}_{G[S]}(V_i) = \mathrm{pa}_G(V_i), \qquad V_i \in S.$$

In particular, every variable $V_i \in C$ keeps the same parent set and hence the same response-function domain. Writing

$$k_i = |\mathrm{pa}_G(V_i)|,$$

a binary variable $V_i \in C$ has $2^{2^{k_i}}$ response types in both graphs. The full and reduced programs therefore use the same joint response-type space and the same decision vector q.

We next show that the fixed conditional distributions of the retained variables outside C are unchanged. For every $V_i \in S \setminus C$, parental closure ensures that all variables in $\mathrm{pa}_G(V_i)$ remain in S. By (S3), every assignment to these parents has positive probability, so the corresponding conditional distribution is well defined. Because P_{core} is the marginal of P on S,

$$P_{\mathrm{core}}\big(V_i \mid V_{\mathrm{pa}_G(V_i)}\big) = P\big(V_i \mid V_{\mathrm{pa}_G(V_i)}\big)\,.$$

Thus, every retained variable outside C uses the same fixed conditional distribution in the full and reduced programs.

We now account for the removed decorations. Let $D = V \setminus S$ denote the set of decorations. Because $C \subseteq S$, every decoration lies outside C. For each decoration $V_d \in D$, let

$$\theta_d\big(v_d \mid v_{\mathrm{pa}_G(V_d)}\big)$$

denote its fixed outside-C conditional distribution, and define

$$\Lambda(v_D \mid v_S) = \prod_{V_d \in D} \theta_d\big(v_d \mid v_{\mathrm{pa}_G(V_d)}\big)\,.$$

Here, v_S and v_D denote assignments to the retained variables and decorations, respectively. The factor Λ does not depend on the joint response-type distribution q.

Parental closure implies that no decoration is an observed parent of a retained variable. Decorations may depend on retained variables and on other decorations, but the directed subgraph on D remains acyclic. Summing the variables in D in reverse topological order therefore gives

$$\sum_{v_D} \Lambda(v_D \mid v_S) = 1$$

for every fixed assignment v_S.

Let P_q^G denote the observational distribution induced on the full graph G by response-type distribution q together with the fixed outside-C conditional distributions. Let $P_q^{G[S]}$ denote the corresponding distribution on the reduced graph. The preceding observations imply

$$P_q^G(v_S, v_D) = P_q^{G[S]}(v_S)\,\Lambda(v_D \mid v_S)$$

for every q. This factorization is the key step in the reduction.

We next compare the base feasible sets. The base regime retains the four cells of $P(X, Y)$. Summing the factorization over assignments consistent with $X = x$ and $Y = y$, and using

$$\sum_{v_D} \Lambda(v_D \mid v_S) = 1,$$

shows that each base probability is the same linear function of q in the full and reduced graphs. The observed right-hand sides also agree because marginalizing the decorations does not change the treatment–outcome distribution:

$$P(X, Y) = P_{\mathrm{core}}(X, Y).$$

Hence,

$$\mathcal{F}_{\mathrm{base}}(G, P) = \mathcal{F}_{\mathrm{base}}(G[S], P_{\mathrm{core}})\,.$$

Whenever these feasible sets are nonempty, the base lower and upper endpoints are therefore identical.

We now compare the full feasible sets. If $q \in \mathcal{F}_{\mathrm{full}}(G, P)$, then q reproduces the full observational distribution $P(V)$. Marginalizing over v_D shows that the same q reproduces P_{core}. Hence,

$$\mathcal{F}_{\mathrm{full}}(G, P) \subseteq \mathcal{F}_{\mathrm{full}}(G[S], P_{\mathrm{core}})\,.$$

For the reverse inclusion, condition (S4) provides at least one

$$\widetilde{q} \in \mathcal{F}_{\mathrm{full}}(G, P).$$

Applying the factorization above to $\widetilde{q}$ gives

$$P(v_S, v_D) = P_{\text{core}}(v_S)\,\Lambda(v_D \mid v_S).$$

Now take any

$$q \in \mathcal{F}_{\text{full}}(G[S], P_{\text{core}})\,.$$

Because this q reproduces P_{core}, the same factorization gives

$$P_q^G(v_S, v_D) = P_{\text{core}}(v_S)\,\Lambda(v_D \mid v_S) = P(v_S, v_D).$$

Thus, q also reproduces the full observational distribution, so

$$\mathcal{F}_{\text{full}}(G[S], P_{\text{core}}) \subseteq \mathcal{F}_{\text{full}}(G, P).$$

Combining the two inclusions yields

$$\mathcal{F}_{\text{full}}(G, P) = \mathcal{F}_{\text{full}}(G[S], P_{\text{core}})\,.$$

It remains to compare the causal-query objective. Under $\mathrm{do}(X = 1)$, the intervention fixes $X = 1$ in both graphs. Since the intervention only removes incoming directed edges to X, parental closure of the retained variables is preserved, and the directed subgraph on D remains acyclic. Because $Y \in S$, marginalizing over the decorations again removes the factor Λ. Every joint response type therefore makes the same contribution to

$$P(Y = 1 \mid \mathrm{do}(X = 1))$$

in the full and reduced graphs. Hence, the causal-query objective vector c is identical in both programs.

The full and reduced programs therefore optimize the same linear objective $c^\top q$ over identical base and full feasible sets. Their four exact endpoints are consequently identical:

$$L_{\text{base}}, \qquad U_{\text{base}}, \qquad L_{\text{full}}, \qquad U_{\text{full}}.$$

The identified-set widths are therefore identical in both regimes, which implies

$$\tau(G, P, Q) = \tau(G[S], P_{\text{core}}, Q)\,.$$

Finally, condition (S3) is required only for retained variables outside C: it ensures that their fixed conditional distributions are determined from the observational distribution and remain unchanged after reduction. The conditional distributions of the removed decorations enter only through Λ and disappear after summing over v_D, so the reduction argument does not require a separate support condition for the decorations.

Benchmark corollary. For the benchmark, the causal-core node set is

$$S^* = \{X, Y\} \cup C \cup \mathrm{an}_G(X) \cup \mathrm{an}_G(Y) \cup \mathrm{pa}_G(C),$$

where $\mathrm{an}_G(\cdot)$ denotes observed ancestors in G and $\mathrm{pa}_G(C)$ denotes the observed parents of variables in C. By construction, S^* contains C, X, and Y, and therefore satisfies condition (S1) of the reduction theorem.

The set S^* satisfies condition (S2) exactly when it is parentally closed, or equivalently when no decoration in $V \setminus S^*$ has a directed edge into a variable in S^*. Under Assumption S, condition (S3) also holds for every retained variable outside C. Finally, every SCM-generated benchmark instance satisfies condition (S4), because its generating response-type distribution is feasible for the corresponding full-regime bounding program.

Why $\mathrm{pa}_G(C)$ is retained. Retaining $\mathrm{pa}_G(C)$ preserves the observed parent set, and therefore the response-function domain, of every variable in C. A binary variable $V_i \in C$ with k_i observed parents has $2^{2^{k_i}}$ possible response types. Removing one of its observed parents therefore changes its response-function domain and, in general, the joint response-type space of the bounding programs.

Consider instead the simpler ancestral reduction

$$\{X, Y\} \cup C \cup \mathrm{an}_G(X) \cup \mathrm{an}_G(Y).$$

If a variable in C has an observed parent outside $\mathrm{an}_G(X) \cup \mathrm{an}_G(Y)$, this reduction removes that parent while retaining its child in C. The retained set is then not parentally closed, so the reduction theorem no longer guarantees preservation of the exact bounds.

This occurs in 12 graph structures in the shape-level diagnostic collection described below. We evaluated 500 sampled mechanisms for each structure. One structure exhibits a discrepancy of 0.149, confirmed with the reference exact solver; eight show no discrepancy above the numerical-equivalence tolerance; and three produce no finite potency label under the ancestral reduction. Thus, removing an observed parent does not necessarily change the bounds for every instance, but exact preservation is no longer guaranteed. Accordingly, we retain $\mathrm{pa}_G(C)$ and treat parental closure as a sufficient, rather than necessary, condition for causal-core reduction.

Numerical validation and scope. We validate the causal-core reduction on two three-to-five-variable graph collections. The shape-level diagnostic collection contains 457 full graph topologies, where graphs remain distinct when their full topologies differ even if they reduce to the same causal core. For each comparison, the reported discrepancy is the absolute difference between the corresponding endpoint or potency computed on the full graph and on its reduced causal core.

We evaluate four SCMs for each of the 258 structures that the simpler ancestral reduction would modify, giving 1,032 graph–SCM pairs. Under the benchmark causal-core definition, 246 of these structures undergo a nontrivial reduction, yielding 984 full-versus-core comparisons. All $246/246$ reductions are parentally closed, so the reduction theorem predicts exact agreement. The largest absolute potency discrepancy is 8.9×10^{-16}.

The core-first benchmark used for the primary evaluation contains 440 representatives with distinct causal cores. Its 25 representatives containing decorations undergo a nontrivial reduction and are evaluated on four SCMs each, producing 100 additional full-versus-core comparisons. All $25/25$ reductions are parentally closed, and the largest absolute potency discrepancy is 5.6×10^{-16}.

Because the theorem establishes equality of all four exact endpoints, we also compare them directly. In the shape-level collection, the largest absolute discrepancies are 5.6×10^{-16} for L_{base}, 7.8×10^{-16} for U_{base}, 2.2×10^{-15} for L_{full}, and 2.4×10^{-15} for U_{full}. In the core-first collection, the corresponding discrepancies are 3.3×10^{-16}, 3.3×10^{-16}, 1.7×10^{-15}, and 1.5×10^{-15}. The largest observed endpoint discrepancy, 2.4×10^{-15}, is more than five orders of magnitude below the 10^{-9} numerical-equivalence tolerance. The observed differences are therefore consistent with finite-precision numerical variation rather than changes in the identified-set endpoints.

As a broader stress check, we evaluate 16,689 graph–SCM pairs generated from denser ADMGs with three to seven observed variables, outside the benchmark generator settings. We find no counterexample satisfying parental closure, and the largest observed potency discrepancy is 2.2×10^{-15}. These results extend the numerical validation beyond the graph distributions used for the primary benchmark. Under the stated conditions, the theorem predicts exact equality, while the observed discrepancies remain at the same negligible numerical scale.

D Model and evaluation details

Predictive metrics. For held-out labels $\{\tau_j\}_{j=1}^N$ and predictions $\{\hat{\tau}_j\}_{j=1}^N$, we use

$$R^2 = 1 - \frac{\sum_{j=1}^N (\tau_j - \hat{\tau}_j)^2}{\sum_{j=1}^N (\tau_j - \bar{\tau})^2}, \qquad \bar{\tau} = \frac{1}{N}\sum_{j=1}^N \tau_j,$$
$$\mathrm{MAE} = \frac{1}{N}\sum_{j=1}^N |\tau_j - \hat{\tau}_j|. \tag{12}$$

Held-out R^2 is the primary metric for the synthetic benchmarks. For the held-out sets used here, $R^2 = 1$ indicates exact prediction, $R^2 = 0$ matches a constant predictor equal to the held-out mean, and $R^2 < 0$ indicates worse performance than that predictor.

For the empirical suite, we report per-query absolute error and dataset-macro MAE. We first average the absolute errors within each dataset and then average these dataset-level errors with equal

weight across datasets. For real-network screening, we report ROC-AUC and AUPRC for the binary targets $\tau > 10^{-6}$ and $\tau > 0.05$, together with the corresponding positive-class prevalence. The threshold 10^{-6} operationalizes positive potency while excluding numerical-zero cases, whereas 0.05 requires an absolute reduction of at least 0.05 in identified-set width and provides a more substantive screening target. The two thresholds therefore evaluate complementary notions of whether auxiliary observational information meaningfully tightens the causal bounds.

Training configuration. For both the three-to-five-variable and exact six-variable benchmarks, PotencyGNN uses hidden dimension $d = 64$ and $R = 3$ message-passing layers. The prediction head contains two hidden layers of width 64, giving the model 67,841 trainable parameters. We optimize the objective in Equation 6 for 220 epochs using Adam with learning rate 3×10^{-3}, weight decay 10^{-4}, minibatch size 512, and a cosine learning-rate schedule. The training schedule is fixed, with no early stopping or held-out checkpoint selection. Predictions are evaluated directly without clipping or projection.

Comparator configurations. The GBM uses aggregated observational and structural features: a 60-dimensional vector for the three-to-five-variable benchmark and a 73-dimensional vector for the exact six-variable benchmark. The fixed GBM is a histogram-based gradient-boosted regressor with at most 300 boosting iterations; a separately tuned configuration is selected using only inner validation cores. GBM fits retain scikit-learn's default `early_stopping='auto'` with `random_state=0`; at these training-set sizes, early stopping uses an internal row-level validation split drawn exclusively from the outer-training data and therefore does not use outer held-out rows or labels.

The flat MLP receives a fixed-order concatenation of the padded node features, directed and bidirected adjacency representations, node mask, global context, and one-hot indicators for the treatment and outcome. On the three-to-five-variable benchmark, it has two ReLU hidden layers of width 200 and 67,201 trainable parameters, closely matching PotencyGNN's parameter count while using no relational weight sharing.

GIN uses three message-passing layers on a symmetrized, untyped support graph formed from the directed and bidirected edges. It uses the same node evidence profiles, global context, and ordered treatment–outcome query information as PotencyGNN and has 42,884 trainable parameters. DeepSets applies a shared node encoder followed by masked mean pooling and therefore does not use graph connectivity. PNA uses the same symmetrized, untyped support graph as GIN and combines mean, maximum, minimum, and standard-deviation aggregation with identity, amplification, and attenuation degree scalers. GRIT applies attention over all valid ordered node pairs and represents directed, reverse-directed, and bidirected relations through pairwise relation indicators and typed relative random-walk features.

All neural comparators use the same node-level observational features, global context, training objective, and core-disjoint training and held-out instances as PotencyGNN, while retaining the graph representation specific to each architecture. They also receive the same ordered treatment–outcome query information through their corresponding input or readout. The same comparator models are evaluated on the empirical suite in Appendix E (Table 9).

Flat-MLP baseline results. The flat MLP achieves mean held-out $R^2 = 0.613$, corresponding to $\Delta R^2 = -0.216$ relative to PotencyGNN (Table 4). PotencyGNN's advantage ranges from 0.194 to 0.238 across the three core-disjoint splits. Given the closely matched parameter counts, these results are consistent with a benefit from explicitly representing relational structure, although parameter matching does not imply equal effective capacity.

Hyperparameter selection. For each benchmark and outer core-disjoint split, hyperparameters for the tuned comparators are selected using only a further partition of the outer-training cores. This inner partition contains 264 fitting and 66 validation cores for the three-to-five-variable benchmark, and 206 fitting and 52 validation cores for the exact-six-variable benchmark. We evaluate at most 12 configurations per model using initialization seeds $0, 1$ and select the configuration with the highest mean validation R^2, breaking ties in favor of fewer parameters and then a prespecified configuration order. The selected configuration is then refitted on all outer-training cores using seeds $0, 1, 2, 3, 4$ for the neural comparators and a single fit with `random_state=0` for GBM, with

all refits evaluated on the untouched held-out cores; no held-out data are used for hyperparameter selection. PotencyGNN and the fixed GBM, flat-MLP, and GIN configurations are prespecified and are not selected or modified through this procedure.

The tuned comparators therefore receive a training-side hyperparameter search that PotencyGNN does not. Conversely, PotencyGNN was designed specifically for this task, whereas the comparator architectures were adapted from published designs. Accordingly, these experiments compare predictive performance under the stated model-selection protocols rather than isolating architecture as a single controlled factor. We therefore interpret the results as evidence of PotencyGNN's competitiveness against strong baselines.

Unconstrained point predictions. By definition, realized potency satisfies $0 \leq \tau \leq w_{\text{base}}$, whereas PotencyGNN uses an unconstrained scalar regression head and does not project its outputs onto this interval. Across the held-out predictions included in this diagnostic, 0.50% are negative, with a largest absolute violation of 4.4×10^{-2}. Projecting predictions onto $[0, w_{\text{base}}]$ changes benchmark R^2 by at most 0.0003 and mean absolute error by at most 6.6×10^{-5}, indicating that these violations have negligible effect on the reported benchmark performance.

Out-of-range predictions become more consequential under the far-out distribution shift examined in the large-response-space stress test of Appendix H. Projection increases R^2 by 0.034 at $K = 131{,}072$ and by 0.003 at $K = 262{,}144$, but does not alter the observed deterioration at these scales. All reported results therefore use the raw outputs of the trained model; projection is considered here only as a diagnostic.

Scope of the exact-runtime comparison. The exact comparator is the response-function linear-programming pipeline used throughout this paper, evaluated on the same inputs and under the same thread budget as PotencyGNN. Specialized methods can reduce the computational cost of related causal-bound problems (Shridharan & Iyengar, 2023a;b; Arroyo et al., 2025), but these methods are not included in our runtime evaluation. Accordingly, the reported speedups are specific to the exact pipeline used in our experiments.

Cross-device sensitivity of the tuned comparators. GRIT and PNA were tuned and evaluated on GPU, whereas the remaining models were evaluated on CPU (Table 1). For split 7 of the three-to-five-variable benchmark, paired CPU–GPU validation results are available for all 12 configurations of GRIT and PNA using two initialization seeds. Both devices selected the same configuration for each model. At the selected configurations, the mean validation R^2 differed by 0.0013 for GRIT and 0.0090 for PNA. Across all evaluated configurations, the largest absolute validation-R^2 discrepancy was 0.042 for GRIT and 0.51 for PNA; the latter occurred for a configuration selected by neither device. PotencyGNN showed a maximum paired CPU–GPU R^2 discrepancy of 0.0067 across 15 fits. Thus, for split 7, device choice did not change the configuration selected for either tuned comparator.

GBM tuning and comparison. On the three-to-five-variable benchmark, PotencyGNN achieves mean held-out $R^2 = 0.829$, compared with 0.709 for the fixed GBM, a difference of 0.120. Training-side hyperparameter selection increases the GBM mean R^2 to 0.717 on this benchmark, while leaving performance on the exact-six-variable benchmark essentially unchanged (0.643 to 0.644). Thus, tuning does not materially alter the PotencyGNN–GBM comparison. The structural-coverage and fixed-budget experiments use the prespecified fixed GBM configuration.

Coverage sweep and fixed-budget analysis. With 60 labels per core, the mean paired PotencyGNN-minus-fixed-GBM R^2 difference changes from -0.042, -0.024, and -0.039 at 32 training cores to $+0.044$, $+0.059$, and $+0.120$ at 64 cores for splits 7, 11, and 13, respectively. Thus, the difference changes sign between 32 and 64 cores on every split, giving the crossover bracket reported in Section 5.3. Under the matched label budget, expanding from four training cores to full structural coverage increases R^2 by 1.042, 0.773, and 1.035 for PotencyGNN, compared with 0.462, 0.417, and 0.488 for the fixed GBM on the same splits.

Controlled variants. The no-propagation variant sets $R = 0$ and therefore has 17,921 parameters. The no-edge variant retains the three update layers and their affine biases but removes all directed

and bidirected adjacency, preserving 67,841 parameters while preventing communication between nodes. The symmetrized-aggregate variant preserves the parameter count but supplies both directed channels with the same sum over incoming and outgoing neighbors, while retaining the bidirected channel and separate $d_i^{\text{in}}, d_i^{\text{out}}$ features. It therefore tests direction-specific aggregation, not removal of all directional information.

Ablation results. All variants are evaluated on the same core-disjoint training and held-out instances, with five initializations per split (Table 4). Removing all graph edges while retaining the full parameter count reduces mean R^2 from 0.829 to 0.277, providing strong evidence that relational propagation is important for transfer to unseen causal cores. Symmetrizing the directed-neighbor aggregates reduces mean R^2 to 0.764, supporting direction-specific aggregation while directed degree features remain available. The no-propagation variant reaches $R^2 = 0.571$, although this comparison is not capacity matched because setting $R = 0$ substantially reduces the parameter count.

The feature ablations have much smaller effects: removing the ρ_i^{xy} profile yields mean $R^2 = 0.823$, while removing the global context $\mathbf{g}$ yields $R^2 = 0.833$. These results provide little evidence that either feature is individually necessary on this benchmark, although they do not rule out redundancy or interactions with the remaining inputs. We retain the complete prespecified model for all headline experiments. Overall, the ablations support graph propagation and direction-specific aggregation. Because the symmetrized-aggregate variant retains directed degree features and the bidirected channel, it neither removes all directional information nor isolates the bidirected relation; matching nominal parameter counts does not guarantee equal effective capacity.

Table 4: Architectural ablations on the three-to-five-variable benchmark. Mean R^2 averages the three split-level means, each computed over five initializations. ΔR^2 is relative to PotencyGNN.

Model or variant	Parameters	Mean R^2	ΔR^2
PotencyGNN	67,841	0.829	—
Flat MLP	67,201	0.613	-0.216
Symmetrized directed aggregates	67,841	0.764	-0.065
Without ρ_i^{xy} profile	67,841	0.823	-0.006
Without global context $\mathbf{g}$	67,841	0.833	$+0.004$
No propagation ($R = 0$)	17,921	0.571	-0.258
No graph edges	67,841	0.277	-0.552

Repeated runs and reporting. Architecture comparisons use split seeds $7, 11, 13$ and model seeds $0, \ldots, 4$. For neural models, within-split means and population standard deviations over five seeds are averaged equally across splits; GBM uses one fit per split with `random_state=0`. Model-difference intervals are 95% percentile intervals from 10,000 paired core-bootstrap replicates per split using seed-mean metrics, without cross-split pooling. Structural-coverage experiments report their subsets and seeds separately; Appendix B specifies the exact-six protocol.

Computing environment. The primary benchmark experiments were run under Windows 11 Education (10.0.26200) on an AMD Ryzen 7 5800X CPU with an NVIDIA RTX 3060 GPU. Predictive benchmark results used CPU execution for PotencyGNN, GBM, flat MLP, DeepSets, and GIN, and GPU execution for GRIT and PNA; certificate training was also performed on GPU. For the timing comparison, PotencyGNN, the exact LP pipeline, and the certificate path were evaluated on a single CPU thread with CUDA disabled, so the reported speedups do not rely on GPU acceleration. The Windows benchmark environment used Python 3.10.9, NumPy 2.2.6, SciPy 1.15.3, scikit-learn 1.7.2, and PyTorch 2.6.0. LP computations and probability-table construction used `float64`, whereas neural model training and inference used `float32`.

Controlled timing protocol. All timed paths begin from the same graph, dense observational table, and ordered query $(G, P(V), X, Y)$; SCM sampling and construction of $P(V)$ are excluded. We sample 64 causal cores without replacement using seed 424242 and evaluate eight distributions per core, yielding 512 inputs. For batch-one timing, we take the median of three repetitions for each input and summarize the resulting 512 paired measurements. Ten complete inputs are used for warm-up before measurement.

Table 5: One-thread CPU timing under the controlled protocol from common $(G, P(V), X, Y)$ inputs. Entries are milliseconds per query. Batch-one columns report the median, IQR, and 90th percentile across 512 per-input medians. Batch-256 columns report the corresponding statistics across 20 measured batch times after division by 256. Cached rows reuse graph-specific LP structure; cold totals include its construction. Totals are timed directly, and ratios are computed from unrounded medians; consequently, sums or ratios reconstructed from the rounded entries may differ slightly.

		Batch one		Batch-256 throughput	
Pipeline	Component	Median [IQR]	P90	Median [IQR]	P90
Exact, cached	LP construction	0.153 [0.127, 0.176]	0.184	0.158 [0.156, 0.161]	0.163
	Base lower solve	1.502 [1.330, 1.834]	2.120	1.618 [1.549, 1.671]	1.683
	Base upper solve	1.500 [1.323, 1.834]	2.169	1.617 [1.551, 1.683]	1.699
	Full lower solve	1.789 [1.487, 2.103]	2.802	1.942 [1.837, 2.045]	2.075
	Full upper solve	1.804 [1.530, 2.142]	2.913	1.993 [1.886, 2.109]	2.145
	Total	6.776 [5.877, 8.200]	10.279	7.346 [6.993, 7.697]	7.846
Exact, cold	Graph-skeleton construction	11.346 [4.195, 17.071]	23.050	—	—
	Total	18.789 [9.567, 24.170]	33.287	—	—
PotencyGNN	Feature and tensor construction	0.349 [0.342, 0.364]	0.387	0.270 [0.267, 0.277]	0.294
	Forward pass	0.475 [0.451, 0.526]	0.568	0.0111 [0.0108, 0.0114]	0.0117
	Total	0.824 [0.794, 0.891]	0.962	0.281 [0.278, 0.287]	0.306
Certificate, cached	LP representation and tensors	0.193 [0.168, 0.214]	0.226	0.143 [0.140, 0.149]	0.158
	Base lower solve	1.517 [1.343, 1.851]	2.144	1.614 [1.511, 1.625]	1.662
	Base upper solve	1.501 [1.329, 1.847]	2.193	1.648 [1.557, 1.685]	1.753
	Network forward pass	0.825 [0.810, 0.858]	0.913	0.590 [0.585, 0.599]	0.606
	Repair and intersection	0.117 [0.116, 0.119]	0.124	0.0200 [0.0198, 0.0204]	0.0206
	Total	4.195 [3.817, 4.898]	5.592	3.977 [3.852, 4.091]	4.151
Certificate, cold	Graph-skeleton construction	11.321 [4.166, 17.102]	23.005	—	—
	Total	15.866 [7.756, 21.389]	28.649	—	—

The cached exact and certificate paths reuse one graph-specific LP skeleton per core, whereas the corresponding cold paths include its construction. PotencyGNN reconstructs its features and structural tensors for every query, and this cost is included in its reported latency. File I/O, model construction or loading, validation, and artifact serialization are excluded from all paths. Neural timing uses fixed seed-0 model weights with the reported architectures; these measurements are used only to characterize execution cost, and no predictive-accuracy result is derived from them. The batch-256 throughput supplement uses two fixed batches, with three warm-up and ten recorded repetitions per batch.

The primary cached batch-one exact-to-PotencyGNN ratio is $8.22\times$ (95% grouped-core bootstrap CI $[7.80, 8.77]$). Including graph-skeleton construction gives a cold ratio of $22.80\times$ (95% CI $[17.35, 24.63]$). At batch size 256, the separately measured throughput ratio is $26.17\times$. We do not report a grouped-core interval for this throughput supplement because it uses two fixed batches rather than the 512 core-indexed paired measurements used for the batch-one analysis. Certificate latency is reported in absolute terms because the certificate computes a conservative lower bound rather than the exact potency value.

Controlled response-space scaling. We complement the broader large-response-space stress test of Section 5.5 with a controlled experiment that varies the joint response-type count K while holding the number of observed variables, clique size, and LP constraint counts fixed. Specifically, we fix $n = 5$, $|C| = 3$, two variables outside C, and 33 full-regime and five base-regime constraint rows. For each $K \in 16, 32, 64, 128, 256, 512, 1024, 2048$, we evaluate six causal cores disjoint from the primary benchmark and generate 32 SCMs per core after applying the benchmark nondegeneracy screen. The corresponding clique-indegree multisets are $(0, 0, 1)$, $(0, 1, 1)$, $(1, 1, 1)$, $(0, 1, 2)$, $(1, 1, 2)$, $(0, 2, 2)$, $(1, 2, 2)$, and $(0, 1, 3)$. All 1,536 retained distributions satisfy Assumption S and pass exact-solver validation.

Table 6: Controlled response-space scaling at fixed node, clique, and constraint counts. Each row contains 192 instances from six causal cores disjoint from the primary benchmark. Latencies are batch-one medians in milliseconds. Ratios compare cached exact evaluation with PotencyGNN and include grouped-core 95% bootstrap intervals. Prediction metrics are averaged over three model seeds. Rows marked † evaluate zero-shot behavior beyond the K range used for training. This controlled experiment complements the broader large-response-space stress test in Section 5.5, which evaluates a different collection of causal structures over a wider range of K.

K	Exact cached	Exact cold	PotencyGNN	Certificate cached	Exact/GNN [95% CI]	MAE	R^2
16	4.386	6.679	0.804	3.274	5.45 [5.41, 5.50]	0.039	0.710
32	4.923	9.342	0.784	3.343	6.28 [6.06, 6.41]	0.028	0.871
64	5.946	11.805	0.782	3.772	7.61 [7.42, 7.68]	0.042	0.742
128	7.085	24.318	0.819	4.255	8.65 [8.19, 8.84]	0.033	0.874
256	10.068	33.297	0.883	5.460	11.40 [11.04, 12.02]	0.024	0.794
512†	14.635	83.439	0.951	—	15.39 [15.17, 15.55]	0.039	0.702
1024†	25.915	119.067	0.958	—	27.05 [26.60, 27.19]	0.041	0.739
2048†	41.039	306.579	0.961	—	42.69 [42.03, 43.19]	0.044	0.787

Timing begins from common $(G, P(V), X, Y)$ inputs on one CPU thread. Cached evaluation reuses the graph-specific LP skeleton, whereas cold evaluation includes its construction. The K groups are interleaved, and latency for each instance is summarized by the median of three repetitions.

Cached exact latency increases from 4.386 to 41.039 ms across the tested range, whereas PotencyGNN latency changes from 0.804 to 0.961 ms. Cold exact latency grows more sharply because it includes LP-skeleton construction. Materialized exact-LP arrays grow from 0.005 to 0.610 MiB and skeleton arrays from 0.006 to 0.440 MiB, excluding solver workspace and Python objects, whereas PotencyGNN's NumPy feature representation remains 492 bytes.

Certificate timings are reported over the evaluated certificate range and provide a complementary view of the cost of certified screening. Although the node, clique, and constraint counts are fixed, increasing K requires different clique indegrees and graph topologies. The experiment therefore characterizes scaling along a controlled K-indexed family rather than varying response-space size in isolation. The rows above $K = 256$ evaluate zero-shot behavior beyond the training range and complement the broader large-response-space stress test in Section 5.5.

Training cost and amortization. PotencyGNN incurs a one-time training cost that is amortized over subsequent potency evaluations. For split 7 and model seed 0, training required 129.2 s. Conditional on $P(V)$ already being available, generating the $19{,}800$ exact training labels required an additional 155.2 s, including 330 graph-skeleton constructions and $19{,}800$ cached exact evaluations. Using the measured batch-one latencies of 6.776 ms for cached exact evaluation and 0.824 ms for PotencyGNN (Table 5), the training cost is amortized after approximately $21{,}700$ subsequent queries when exact labels are already available, or $47{,}800$ queries when label generation is included. Under the batch-256 throughput measurements in Table 5, the corresponding break-even points are approximately $18{,}300$ and $40{,}300$ queries.

E Empirical evaluation details

Query construction and reporting units. The Vitamin A, Lipid, Fertility, and JTPA analyses use the binary instrumental-variable ADMG $Z \to X \to Y$ with a posited $X \leftrightarrow Y$ relation. In the three trial datasets, Z denotes randomized assignment or offer; in Fertility, it denotes the same-sex composition of the first two children. The graph's exclusion restriction and latent-confounding structure are maintained assumptions, not conclusions established by these data. For Sachs, we construct queries by hiding a measured parent in the published consensus signaling graph and applying latent projection to the remaining variables. Three queries under median discretization form the primary Sachs analysis; tercile and mean discretizations provide six sensitivity queries. These rows produce finite base/full LP solutions; their targets are plug-in potencies of $\widehat{P}(V)$ under the specified graph and restricted SCM, not ground-truth effects or evidence of model validity. The suite contains seven primary and twelve sensitivity queries. Multiple queries within a dataset reuse observations or differ

only in coding, subgroup, or discretization. The resulting 19 rows are therefore not independent replicates, and we do not summarize them using pooled R^2.

Sensitivity analyses. Table 7 reports the twelve sensitivity queries as six individual IV queries and two groups of three Sachs queries. For each query, we take the absolute difference between its reference potency and the mean prediction of the 15 PotencyGNN models. Each Sachs row reports the mean of this error over its three queries. These analyses are separate from the seven primary queries used for dataset-macro MAE.

Table 7: Sensitivity analyses for empirical transfer. Reference potencies are computed by the exact bounding programs under each specified graph and empirical distribution. Predictions average the 15 synthetic-trained PotencyGNN models (five initializations for each of splits 7, 11, and 13), without empirical fitting or model selection. Sachs reference potencies are zero at the reported precision for all three queries in each row; their errors average the three query-level absolute errors. Table 8 reports a separate comparison using only the five split-7 models.

Analysis	Reference potency	PotencyGNN error
Lipid, reversed outcome coding	0.3124	0.0203
Fertility, worked $\geq$ 26 weeks	0.0334	0.0052
Fertility, worked 52 weeks	0.0334	0.0103
JTPA, men	0.2042	0.0172
JTPA, women	0.2112	0.0241
JTPA, administrative-spell treatment	0.1801	0.0163
Sachs, tercile discretization	0 for all three	0.0068
Sachs, mean discretization	0 for all three	0.0072

Structural overlap. All positive-potency empirical queries use the binary instrumental-variable core described above. This core is absent from training in split 7 but present in the training sets of splits 11 and 13. We therefore report split 7 separately when assessing transfer to this unseen core: the five split-7 models achieve MAE 0.0169 across the ten positive IV queries and 0.0152 across the four primary IV queries. The 15-model average combines seen- and unseen-core settings and summarizes transfer to empirical distributions rather than generalization to an unseen core. The projected Sachs cores are absent from the 440-core synthetic benchmark, but their reference potencies are zero at reported precision; these queries assess prediction near zero on new structures.

For the ten IV queries, the compliance margin $P(X = 0) - \min_z P(X = 0 \mid Z = z)$ matches the exact plug-in potency to machine precision. This equality is verified for the evaluated tables and is not assumed for other binary IV distributions. A constant-zero predictor likewise matches the Sachs reference values at reported precision. These simple comparators clarify the scope of the empirical results: the suite assesses predictive accuracy on empirical distributions with limited structural diversity, while broader generalization across held-out causal cores is evaluated in Section 5.2.

Relation to experiment-indexed pruning. Maringgele & Etesami (2026) give graphical criteria for certifying that a candidate experiment cannot tighten bounds when the full observational distribution $P(V)$ is already available. Our comparison starts instead with the $P(X, Y)$ constraints and adds the remaining observational constraints to obtain the full regime, while holding the outside-C conditionals fixed. Their criteria therefore do not directly classify zero potency under our base-to-full comparison: information redundant given $P(V)$ may still tighten bounds relative to our base regime. We compute that reduction directly for the published-network queries, 144 of which have $\tau > 10^{-6}$.

Variable definitions and source validation. For JTPA, the primary treatment indicator is the training variable provided in the replication data of Abadie et al. (2002). We adopted this definition after the outcome data had been accessed under earlier variable mappings, so the change was not pre-specified. An alternative indicator reconstructed from recorded training spells is reported in Table 7 as a sensitivity analysis. It is distinct from the earlier mapping that produced a first-stage pattern incompatible with the study design. For Lipid, we calculate the empirical distribution from raw cell counts. As an input check, our solver reproduces the published average-causal-effect bounds when

applied to the three-decimal frequencies reported by Ramsahai & Spirtes (2012). The corresponding bounds from raw counts differ by less than 10^{-3} at each endpoint; all reported Lipid potency results use raw-count inputs.

Table 8: Empirical transfer on the IV core held out from synthetic training in split 7. Entries are absolute errors on the four primary IV queries; Macro averages these four errors. For each neural model, predictions from five split-7 initializations are averaged before computing error. Tuned GBM uses its single split-7 fit. No model is fitted, tuned, or selected using empirical data.

Model	Vitamin A	Lipid	Fertility	JTPA	Macro
PotencyGNN	0.0221	0.0098	0.0052	0.0239	0.0152
GRIT (tuned)	0.0402	0.0460	0.0017	0.0064	0.0236
GBM (tuned)	0.0844	0.0098	0.0032	0.0240	0.0303
PNA (tuned)	0.0605	0.0609	0.0108	0.0740	0.0515

Extended baseline comparison. We evaluate the nine baseline configurations described in Section 5.1 and Appendix D on all 19 empirical queries. Fixed configurations were prespecified, and tuned configurations were selected only on synthetic training-side validation cores; no model was fitted or selected using empirical data. Before empirical evaluation, the retrained models passed checks against their recorded synthetic held-out predictions. Among the evaluated learned models, PotencyGNN has the lowest dataset-macro MAE on the primary analyses (0.0125), followed by tuned GRIT (0.0205) and tuned GIN (0.0271). This descriptive ordering does not hold on every dataset (Table 9).

Split 7 provides a separate comparison on the binary IV core excluded from its training data. On the four primary IV queries, PotencyGNN has lower MAE than tuned GRIT (0.0152 versus 0.0236; Table 8). Across all ten IV queries, including sensitivity variants, tuned GRIT has lower MAE (0.0146 versus 0.0169) (Table 9). These summaries differ in both their query sets and dataset weights: JTPA contributes four of the ten query-level errors but only one of the four primary-query errors, and GRIT is more accurate on all four split-7 JTPA queries. The two summaries therefore do not establish a consistent ordering between the models on this empirical suite.

Table 9: Learned-model errors against exact plug-in potency on the empirical suite. The first four dataset columns give absolute error on one primary query each; Sachs gives MAE over three median-discretization queries. Dataset macro weights the five datasets equally. Split-7 IV MAE averages the ten positive IV queries, including sensitivity variants, using only models whose training excludes the IV core. Errors are computed after averaging predictions: 15 models per neural configuration and three split-specific fits per GBM configuration for the primary columns; five neural models or one GBM fit for the split-7 column. Values are rounded to four decimals.

Model	Vitamin A	Lipid	Fertility	JTPA	Sachs (median)	Dataset macro	Split-7 IV MAE
PotencyGNN	0.0210	0.0062	0.0045	0.0214	0.0091	0.0125	0.0169
GRIT (tuned)	0.0286	0.0449	0.0006	0.0109	0.0175	0.0205	0.0146
GIN (tuned)	0.0432	0.0200	0.0096	0.0516	0.0111	0.0271	0.0335
PNA (tuned)	0.0324	0.0375	0.0053	0.0529	0.0154	0.0287	0.0547
GIN (fixed)	0.0469	0.0217	0.0138	0.0621	0.0165	0.0322	0.0398
GBM (tuned)	0.1105	0.0163	0.0002	0.0095	0.0282	0.0329	0.0249
Flat MLP (fixed)	0.0034	0.0511	0.0000	0.0671	0.0588	0.0361	0.0478
DeepSets (tuned)	0.0035	0.0931	0.0044	0.0705	0.0166	0.0376	0.0642
GBM (fixed)	0.1234	0.0308	0.0015	0.0141	0.0241	0.0388	0.0274
Flat MLP (tuned)	0.0183	0.0644	0.0033	0.0806	0.0497	0.0432	0.0491

Published-network screening. We construct candidates from 21 discrete Bayesian networks in the bnlearn repository (Scutari, 2010). Each candidate selects an ancestrally closed subgraph of at most five variables and an ordered pair (X, Y) for which X is an ancestor of Y. We compute its joint distribution from the published conditional probability tables (CPTs), binarize each variable as state 0 versus all other states, retain the published directed edges, and posit $X \leftrightarrow Y$. The resulting distributions come from network models; binarization and the added confounding relation are maintained assumptions. Of 1,811 candidates, 156 violate Assumption S and 304 of the remaining 1,655 fail the numerical LP-feasibility screen; we exclude these failures without interpreting them

as model incompatibility. Deduplicating the 1,351 candidates with finite base/full solutions leaves 400 queries from 20 networks.

We classify 144 as above 10^{-6} and 256 as at or below it. The cutoff separates near-zero computed values from larger values; it does not define a practically meaningful reduction. In the final 400 labels, the largest value at or below the cutoff is 1.81×10^{-7}, and the smallest above it is 2.50×10^{-5}. Hence any cutoff between these values gives the same classification. We separately report detection of $\tau > 0.05$ to assess larger reductions in identified-set width.

The screening models are trained separately from the main core-disjoint models, using a synthetic training split by graph topology. The *uniform* arm draws base CPT probabilities from $\mathrm{Uniform}(0.15, 0.85)$, the range used by the original synthetic generator. The *calibrated* arm instead draws from a pool of binary-node CPT probabilities taken from the same 21 bnlearn networks, after clipping the pool to $[0.005, 0.995]$ to avoid exact 0 and 1. Thus the calibrated arm uses source-network parameter information, although neither arm is trained on the 400 potency labels. This evaluation does not isolate generalization to unseen causal cores.

Table 10 averages each metric over five model seeds. To assess variation across source networks, we additionally form 2,000 bootstrap replicates, sampling networks with replacement and selecting one model seed per replicate. For pooled R^2, the bootstrap median and 95% percentile interval are $0.450\,[-0.436, 0.780]$ for uniform training and $0.428\,[0.007, 0.867]$ for calibrated training. These bootstrap summaries differ from the across-seed mean and standard deviation reported in the table.

Table 10: Screening on 400 supported published-network queries. The 256 queries with $\tau \le 10^{-6}$ are not necessarily exactly zero. Values are means over five model seeds; $\pm$ gives the population standard deviation across seeds. Pooled R^2 uses all 400 queries, whereas positive-stratum R^2 uses only the 144 queries selected by their exact potency labels and is therefore a conditional regression diagnostic rather than a screening metric.

Metric	Uniform	Calibrated
ROC-AUC, $\tau > 10^{-6}$ (prevalence 0.360)	0.682	0.582
AUPRC, $\tau > 10^{-6}$	0.566	0.491
ROC-AUC, $\tau > 0.05$ (prevalence 0.100)	0.934	0.933
AUPRC, $\tau > 0.05$	0.583	0.582
Pooled R^2	0.451 ± 0.050	0.405 ± 0.064
Positive-stratum R^2	0.774 ± 0.014	0.898 ± 0.026
Mean prediction, $\tau \le 10^{-6}$	0.043	0.042

F Dual-certificate architecture and training

Architecture. The certificate network takes the full-regime LP data $Aq = b$ and objective vector c as input, with $A \in \mathbb{R}^{33\times256}$, $b \in \mathbb{R}^{33}$, and $c \in \mathbb{R}^{256}$. Row 0 enforces $\mathbf{1}^\top q = 1$. Each of the next 2^n rows requires the model to reproduce one joint assignment probability in $P(V)$; unused row positions are zero-padded. The assignments follow a fixed lexicographic order in the stored topological node order, with the first node as the most significant bit. Corresponding positions in A, b, and the predicted dual residuals refer to the same constraint. A mask identifies valid response-type columns and excludes padded columns from pooling, anchor construction, and feasibility repair.

Each valid column j corresponds to a joint response type (Section 3). Its column $A_{:j}$ records its contribution to the constraints, while c_j gives its causal-query value. A shared encoder computes

$$h_j^{(1)} = \mathrm{ReLU}(W_1[A_{:j}; c_j] + \beta_1) \in \mathbb{R}^{192},$$
$$h_j^{(2)} = \mathrm{ReLU}(W_2 h_j^{(1)} + \beta_2) \in \mathbb{R}^{192},$$
$$e_j = W_3 h_j^{(2)} + \beta_3 \in \mathbb{R}^{192}.$$

Masked mean and coordinatewise maximum pooling (Equation 8) yield $h_{\mathrm{LP}} \in \mathbb{R}^{384}$. Concatenating this representation with b and the objective-range anchors gives

$$\xi = [h_{\mathrm{LP}} \,\|\, b \,\|\, \alpha_L \,\|\, \alpha_U] \in \mathbb{R}^{419}.$$

The residual predictor computes

$$\begin{aligned} u^{(1)} &= \mathrm{ReLU}(V_1\xi + \gamma_1) \in \mathbb{R}^{256}, \\ u^{(2)} &= \mathrm{ReLU}(V_2u^{(1)} + \gamma_2) \in \mathbb{R}^{256}, \\ u^{(3)} &= \mathrm{ReLU}(V_3u^{(2)} + \gamma_3) \in \mathbb{R}^{256}, \\ [r_L; r_U] &= V_4u^{(3)} + \gamma_4 \in \mathbb{R}^{66}. \end{aligned}$$

Thus $r_L, r_U \in \mathbb{R}^{33}$. The output has no activation because equality-constrained dual variables are unrestricted in sign. The network has 336,898 trainable parameters and shares no parameters with PotencyGNN.

Soundness under exact arithmetic. Equation 9 gives

$$A^\top \widetilde{y}_L - s_L\mathbf{1} \le c.$$

Because $A^\top e_0 = \mathbf{1}$, the repaired vector satisfies $A^\top y_L^{\mathrm{rep}} \le c$. The upper repair similarly gives $A^\top y_U^{\mathrm{rep}} \ge c$. Hence, for every full-regime feasible $q \ge 0$ with $Aq = b$,

$$L_{\mathrm{dual}} = b^\top y_L^{\mathrm{rep}} \le c^\top q \le b^\top y_U^{\mathrm{rep}} = U_{\mathrm{dual}}.$$

The repaired dual endpoints therefore enclose every value in $\mathcal{I}_{\mathrm{full}}(Q)$. The base interval also encloses this set because $\mathcal{I}_{\mathrm{full}}(Q) \subseteq \mathcal{I}_{\mathrm{base}}(Q)$. Their intersection gives Equation 10, and its inclusion within the base interval gives Equation 11. Network predictions affect the tightness of the repaired interval, but not these exact-arithmetic guarantees.

Finally, every base-feasible q is a probability vector and $\alpha_L \le c_j \le \alpha_U$ for every j. Thus $[L_{\mathrm{base}}, U_{\mathrm{base}}] \subseteq [\alpha_L, \alpha_U]$, so intersecting separately with the objective range cannot tighten the certificate.

Training objective. We train the certificate network to tighten its repaired-dual outer interval without supervising it with full-regime endpoints or optimal dual solutions. For a minibatch of size B, the loss is

$$\mathcal{L}_{\mathrm{cert}} = \frac{1}{B}\sum_{i=1}^{B} \left(U_{\mathrm{dual},i} - L_{\mathrm{dual},i}\right). \tag{13}$$

This is the interval width before intersection with the base bounds. Optimization affects tightness; certificate validity follows from dual-feasibility repair and the separate numerical verification.

Training and evaluation protocol. The study uses the primary benchmark's 440 causal cores and core-disjoint partitions, with 80 independently sampled SCM parameterizations per core. Each of three splits has 330 training cores and 110 held-out cores, yielding 26,400 training and 8,800 held-out instances. We train three independently initialized networks per split, for 79,200 held-out model–instance evaluations in total. The certificate retains the base-regime bounding programs at inference; reference full-regime endpoints are used only to measure containment and tightness on held-out instances.

Models are trained for 400 epochs with Adam, learning rate 10^{-3}, weight decay 10^{-5}, minibatches of 1024, and cosine learning-rate annealing. The schedule is fixed without early stopping or checkpoint selection on held-out data. Network training uses 32-bit GPU arithmetic; the certificate's finite-precision guarantee is established by the separate verification procedure below.

Arithmetic model and scope. After the network predicts candidate dual vectors, we verify its dual bounds using outward enclosures under IEEE 754 binary64 round-to-nearest with gradual underflow. The verifier covers matrix products, feasibility repair, and endpoints. Conditional on nonemptiness of the full feasible set, these bounds apply to the exact-real LP defined by the supplied binary64 arrays; the verifier does not establish full-LP feasibility or cover the population distribution, unrounded probabilities, or reference-solver output.

Canonical base representation. The base regime requires the model-implied probabilities of the four (X, Y) cells to match $P(X, Y)$. Both sets of cell probabilities sum to one, so imposing normalization makes one of the four cell equalities redundant. We therefore retain normalization and three cell equalities; the fourth follows from their sum. This equivalent formulation leaves the feasible response-type distributions, the causal-query objective, and the sharp base interval unchanged.

Outward verification. Equation 9 establishes dual feasibility in exact arithmetic. For the programs represented by the stored binary64 arrays, we instead enclose each dual constraint and objective value outward. The inner-product enclosures account for rounding and underflow (Rump, 2013); the quantities used to form these enclosures are themselves bounded outward.

For an upper-dual candidate $\widetilde{y}_U$, let $t_j^{\mathrm{lo}} \le (A^\top \widetilde{y}_U)_j$ be a verified lower bound for each constraint value. We choose the normalization-row shift to upper-bound $\max\{0, \max_j(c_j - t_j^{\mathrm{lo}})\}$. Because $A^\top e_0 = \mathbf{1}$, the shifted vector is dual feasible. An outward upper bound on its objective then bounds U_{full} by weak duality. The lower endpoint is verified symmetrically.

Certified base quantities. The base regime supplies two bounds with different roles. To intersect intervals, we need a base outer interval that contains the full-regime identified set. We obtain it from verified base dual bounds, widened to account for the rounding discrepancy between the stored base constraints and the corresponding full-regime marginals. To lower-bound the base width, we instead verify two feasible base-regime response-type distributions q^- and q^+ in exact rational arithmetic. Their query-value difference satisfies $c^\top(q^+ - q^-) \le w_{\mathrm{base}}$. Using an outer interval for intersection and a lower width bound for subtraction prevents the certificate from overstating potency.

Finite-precision semantics. Let $\tau_{\mathrm{cert}}^{\mathrm{fp}}$ be the verified lower bound on the base width minus the width of the verified outer interval after intersection. The canonical binary64 base and full programs need not be exactly nested, so this quantity may be negative. Conditional on full-LP feasibility, $\tau_{\mathrm{cert}}^{\mathrm{fp}} > 0$ certifies that much width reduction; a nonpositive value is inconclusive. For screening, we represent such values by $\max\{0, \tau_{\mathrm{cert}}^{\mathrm{fp}}\}$, without treating zero as a positive certificate. The 10^{-5} reporting threshold excludes numerically negligible positive values from the reported screening fraction; it is not required for certificate validity. In the exact-arithmetic formulation, $\tau_{\mathrm{cert}} \ge 0$ follows from Equation 11.

Numerical regression audit. We compared the outward-verified certificates with the earlier numerical evaluation across all 79,200 model–instance evaluations (3 splits $\times$ 3 initializations $\times$ 8,800 held-out instances). The largest change in signed certificate value was 1.19×10^{-14}. No decision at the 10^{-5} reporting threshold or top-decile membership changed: the smallest distances to these boundaries were 3.27×10^{-6} and 1.71×10^{-5}, respectively. Three verified positive certificates fall below the reporting threshold and remain valid positive certificates. The sole reported summary retained from the earlier evaluation is the pre-intersection excess width in Section 5.7; the final post-intersection certificate and screening results use outward verification.

Certificate runtime. Certificate generation is separate from PotencyGNN point prediction and includes base-bound computation and outward verification. Its latency is therefore reported separately. The speedups in Section 5 concern point prediction and do not include certificate computation.

G OUTSIDE-CLIQUE PARENT SUPPORT

Assumption S (outside-clique parent support). For each $V_i \notin C$, every configuration of its observed parents has positive probability under $P(V)$:

$$P(V_{\mathrm{pa}_G(V_i)} = a) > 0 \qquad \text{for all } a \in \{0,1\}^{|\mathrm{pa}_G(V_i)|}.$$

For a variable with no observed parents, this condition holds automatically.

Outside-clique conditional distributions. Assumption S makes each conditional distribution used for $V_i \notin C$ well defined. For $v \in \{0, 1\}$ and every parent configuration a,

$$P(V_i = v \mid V_{\mathrm{pa}_G(V_i)} = a) = \frac{P(V_i = v, V_{\mathrm{pa}_G(V_i)} = a)}{P(V_{\mathrm{pa}_G(V_i)} = a)}.$$

The denominator is positive by Assumption S, so $P(V)$ uniquely determines these distributions. They are fixed in both bounding regimes; apart from these shared kernels, the base program constrains only $P(X, Y)$. For a variable without observed parents, the fixed distribution is $P(V_i = v)$.

Scope when support fails. If $P(V_{\mathrm{pa}_G(V_i)} = a) = 0$, $P(V)$ does not determine the conditional distribution of V_i at a, although that parent configuration may occur under $\mathrm{do}(X = 1)$. The bounding programs assign $P(V_i = 1 \mid V_{\mathrm{pa}_G(V_i)} = a) = 1/2$ in this case. This completion is inactive under Assumption S. When support fails, the resulting bounds may depend on this completion and may exclude query values attainable under other observationally compatible completions. Accordingly, our sharpness claim is restricted to instances satisfying Assumption S.

H LARGE-RESPONSE-SPACE STRESS TEST

Setup. We evaluate 2,242 canonical instances at 14 response-type counts between $K = 16$ and $K = 262{,}144$. The $K = 16$ group contains 162 instances from six causal cores; each of the other 13 values contains 160 instances from 20 cores. For the three-variable clique considered here, the response-type count in Section 3 implies

$$\log_2 K = \sum_{V_i \in C} 2^{|\mathrm{pa}_G(V_i)|}.$$

Consequently, $K = 32{,}768$ is absent because $\log_2(32{,}768) = 15$ cannot be expressed as a sum of three powers of two. The instance set and evaluation protocol were fixed before the reported accuracy results were examined; the stored outputs were subsequently checked for completeness and integrity.

Accuracy and latency across response spaces. Figure 2 shows positive R^2 throughout the $K \leq 256$ training range. Beyond it, the point estimates remain positive at $K = 512$ and $K = 1{,}024$, although the grouped-core interval at $K = 512$ includes zero. At larger response-type counts, R^2 becomes unstable and eventually strongly negative; the increase in absolute error confirms that this is not solely an effect of changing label dispersion. PotencyGNN has higher R^2 than the GBM comparator at every evaluated K, but both models perform poorly at the largest counts. Cached exact latency rises sharply while PotencyGNN latency stays near 1 ms, so low inference cost persists beyond the range of reliable prediction.

Exposure status. Some stress-test instances and causal cores appeared in earlier preliminary evaluations. We therefore classified each evaluated K value, before analyzing the final results, as contextual, partially exposed, or audit-unexposed; these categories are shown in Figure 2 and Table 11. Values at $K \leq 256$ lie within the response-type range used for training, whereas $K > 256$ evaluates zero-shot transfer to larger response-type spaces. At the four audit-unexposed values ($K = 512$, 4,096, 8,192, and 131,072), the R^2 point estimates decline from 0.580 to -6.365. Thus, deterioration at large K is also observed at response-type counts without recorded prior evaluation exposure.

Exposure sensitivities. We assess the effect of documented prior evaluation using three exclusions defined before computing the final results. Sensitivity A excludes instances whose exact potency labels appeared in earlier analyses; B excludes causal cores also used at the same K in the controlled response-type scaling study; and C applies both exclusions. The two studies use different SCM draws and model checkpoints, although some causal cores overlap. The primary analysis retains all 2,242 instances. As Table 12 shows, excluding overlapping cores changes R^2 at $K = 1{,}024$ from 0.668 to 0.644. At $K = 2{,}048$, the estimates range from -0.036 to 0.015 and remain close to zero. The negative results at $K = 65{,}536$ and $K = 262{,}144$ also persist after the applicable exclusions. These analyses therefore leave the observed deterioration at the largest response-type counts intact.

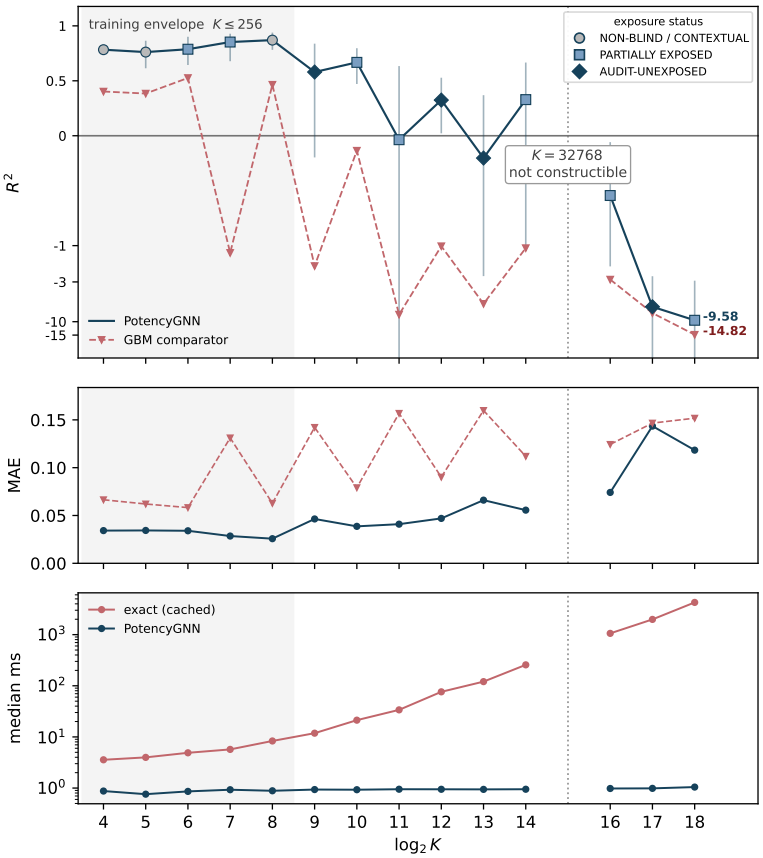

Figure 2: Large-response-space stress test on 2,242 canonical instances. Top: R^2 for PotencyGNN and the prespecified GBM comparator against $\log_2 K$. Marker shapes indicate exposure status; error bars give grouped-core 95% bootstrap intervals where finite. No interval is reported at $K = 16$ (six cores), and the interval at $K = 262{,}144$ is unbounded. The R^2 axis is linear on $[-1, 1]$ and logarithmic in magnitude outside that range; values at $K = 262{,}144$ are labelled. Middle: mean absolute error, shown because changes in label dispersion affect R^2. Bottom: median batch-one latency on a logarithmic axis. Shading marks the training range $K \leq 256$. The curves break at the structurally unavailable $K = 32{,}768$. The controlled scaling study in Table 6 uses different checkpoints and instances.

Timing. Figure 2 shows that median cached-exact latency rises from 3.57 ms at $K = 16$ to 4,265.83 ms at $K = 262{,}144$, while PotencyGNN feature construction and prediction changes from 0.874 ms to 1.046 ms. The median of the matched instance-level latency ratios increases from $4.02\times$ at $K = 16$ to $9.57\times$ at $K = 256$ and $4{,}015\times$ at $K = 262{,}144$. This statistic differs from dividing the two latency medians. At the largest response-type count, $R^2 = -9.579$, so the computational speedup does not imply reliable potency predictions there.

Label-distribution context. The distribution of exact potency changes across response-type counts. From $K = 16$ to $K = 262{,}144$, its standard deviation falls from 0.106 to 0.043, while the fraction of labels at or below 10^{-6} rises from 0 to 0.900; neither pattern is monotone at intermediate K. Lower label variance contributes to the strongly negative R^2 values at the largest response spaces, since R^2 normalizes squared error by label variation. This does not explain away the loss of accuracy: PotencyGNN's MAE also rises from 0.034 to 0.118 between these endpoints. As a retrospective diagnostic, a constant-zero predictor has lower MAE at six of the 14 evaluated K values, all beyond the training range. Because potency is nonnegative, that predictor's MAE equals the mean potency label.

Table 11: Large-response-space results by response-type count. Exposure status is contextual (N), partially exposed (P), or audit-unexposed (A). Here n denotes instances and c causal cores. PotencyGNN metrics average three models; the reported 95% R^2 intervals resample causal cores. No interval is reported at $K = 16$ because this group contains only six cores. The comparator is the prespecified large-K GBM trained on synthetic training-bank instances, distinct from the tuned GBM in Table 1.

K	$\log_2 K$	Status	n	c	PotencyGNN		GBM		ρ	R^2 interval
					R^2	MAE	R^2	MAE		
16	4	N	162	6	0.783	0.034	0.402	0.066	0.883	–
32	5	N	160	20	0.761	0.034	0.384	0.062	0.870	$[0.62, 0.86]$
64	6	P	160	20	0.787	0.034	0.526	0.058	0.898	$[0.65, 0.89]$
128	7	P	160	20	0.853	0.029	-1.246	0.131	0.698	$[0.69, 0.92]$
256	8	N	160	20	0.870	0.026	0.464	0.063	0.950	$[0.79, 0.93]$
512	9	A	160	20	0.580	0.046	-1.852	0.142	0.621	$[-0.19, 0.83]$
1024	10	P	160	20	0.668	0.039	-0.136	0.079	0.795	$[0.48, 0.79]$
2048	11	P	160	20	-0.036	0.041	-8.055	0.157	0.273	$[-29.62, 0.63]$
4096	12	A	160	20	0.325	0.047	-1.016	0.090	0.472	$[0.03, 0.52]$
8192	13	A	160	20	-0.203	0.066	-5.814	0.160	0.213	$[-2.45, 0.36]$
16384	14	P	160	20	0.329	0.056	-1.077	0.112	0.509	$[-1.12, 0.66]$
65536	16	P	160	20	-0.544	0.074	-2.791	0.124	0.543	$[-1.83, -0.07]$
131072	17	A	160	20	-6.365	0.143	-7.698	0.147	-0.221	$[-33.28, -2.59]$
262144	18	P	160	20	-9.579	0.118	-14.817	0.152	-0.058	$[-4.67 \times 10^{29}, -2.96]$

Table 12: Exposure sensitivities for PotencyGNN, reported as mean R^2 over three models. A excludes previously reported instances, B excludes overlapping causal cores, and C applies both exclusions. A dash indicates that the sensitivity excludes no instances at that K, so its result equals the primary value. NC indicates that all instances are excluded and R^2 cannot be computed.

K	Primary	A	B	C
16	0.783	0.782	NC	NC
32	0.761	0.802	0.729	0.775
64	0.787	–	0.759	0.759
128	0.853	–	0.861	0.861
256	0.870	0.876	0.868	0.875
512	0.580	–	–	–
1024	0.668	–	0.644	0.644
2048	-0.036	-0.030	0.010	0.015
4096	0.325	–	–	–
8192	-0.203	–	–	–
16384	0.329	0.334	–	0.334
65536	-0.544	-0.515	–	-0.515
131072	-6.365	–	–	–
262144	-9.579	-9.442	–	-9.442