DEFEASIBLE AND DEONTIC PROBABILISTIC CIRCUITS: SEMANTIC TOPOLOGY FOR BELIEF REVISION AND NORMATIVE AUDIT

Anonymous authors
Paper under double-blind review

ABSTRACT

A model that learns from what people do can predict what they will do next, but it cannot tell whether what they do is allowed or not. Nor does it have a principled way to retract a conclusion when an exception arrives without disturbing everything else it believes. To address this, we introduce Defeasible Probabilistic Circuits (DPCs) for structured belief revision and Deontic Probabilistic Circuits (DePCs) for normative auditing. These models designate selected gated sum nodes of a conditional probabilistic circuit as defeasible or deontic (audited) nodes, preserving exact, linear-time inference. To support defeasible reasoning—where previous conclusions can be overridden by new evidence—a DPC structures a node so one branch is the default and the rest are prioritized exceptions. It compiles any learned softmax gate exactly into this ordered-rule form without retraining. A new exception can then be inserted without altering higher-priority weights, accompanied by an evidence-specific certificate bounding its effect. To support deontic reasoning—evaluating what is allowed versus forbidden—a DePC designates branches as either compliant or violating. It pairs the learned descriptive distribution with a policy-supplied normative reference, exactly computing severity-weighted departures and using a gate-wise audit budget to localize policy violations. Across two physical-reasoning and two clinical environments, DPCs and DePCs improve update and action ranking, respectively, over comparably sized conditional circuits. Although trained solely from observations and without ranking supervision, they also outperform GPT-5.6 Sol, Qwen3.8-27B, and Qwen3-32B under zero-shot evaluation.

1 INTRODUCTION

Consider the following scenario. A nurse is preparing a high-alert medication. The hospital requires a second nurse to check the dose independently before it is given. On a busy night shift, the check is often skipped. A model trained on the ward's records learns this faithfully. Asked what happens next, it predicts that the check will be skipped, and it is correct. What the model cannot tell us is that the event it just predicted is a *violation*. The records say what nurses do. They say nothing about what nurses are *supposed to do*, and collecting more records from the same ward will not change that. Consider a second scenario. A robot sees a cup on a table and concludes that the cup is stable. Then someone bumps the table. The robot should give up one conclusion, that the cup is stable, and keep the rest of what it believes about the scene. If it averages the two readings, or lets the bump nudge every belief it holds, it has not revised its beliefs. It has blurred them.

Both examples are old problems in knowledge representation. *Defeasible reasoning* studies conclusions that hold by default but yield to exceptions: birds fly, but penguins do not (Reiter, 1980; Nute, 2003; Pollock, 1987). The challenge is not that conclusions change, since Bayesian conditioning already does, but that the change should be structured. An exception should override its default without disturbing unrelated conclusions (Grosof, 1991). *Deontic reasoning* studies the gap between what is and what ought to be (von Wright, 1951). Violations are common and central to audit, and can create further obligations: if the double-check was skipped, the dose must be re-verified before administration, an obligation that exists only because the first was broken. Standard deontic logic famously cannot represent such situations consistently (Chisholm, 1963; Carmo & Jones, 2002).

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.

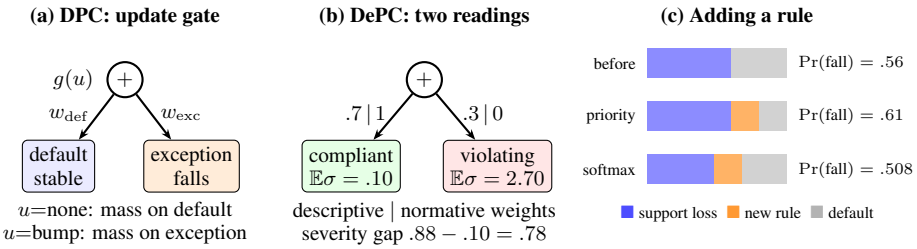

Figure 1: Branch roles give a gated circuit three exact operations. (a) An exogenous update shifts mass from a DPC default to an exception. (b) A DEPC uses learned weights descriptively and compliant branches normatively; their expected-severity difference gives an exact audit (Example 11). (c) Priority insertion takes the new rule's mass from the default while preserving the stronger support-loss rule; a softmax append takes mass from every branch (Section 3).

Standard probabilistic and neural models face both problems. First, a model trained on observations learns how often events occur, and frequency carries no information about what is permitted. The ward's records would be identical whether skipping the check were allowed or forbidden, so the rule must come from outside the data, from a protocol or expert. We show that this is unavoidable: no estimator based on observed behavior alone can recover a norm (Proposition 4). Second, when such a model receives new evidence, every correlated belief shifts according to the statistical dependencies it has learned. Nothing in the model distinguishes a default from the exception that overrides it, so it cannot guarantee that the bump revises the belief about the cup's stability and leaves unrelated beliefs about the scene intact. Neither limitation is a matter of capacity. Language models can often state such rules, but they maintain no probabilistic model of a specific environment against which a rule can be checked, and in our experiments they rank the effects of updates below much smaller circuits trained on that environment (Section 5).

In this paper, we address both problems within probabilistic circuits (PCs), a family of tractable models that compute exact marginals in time linear in their size (Poon & Domingos, 2011; Choi et al., 2020). Our approach adds two ingredients. The first is *update gates*: sum nodes whose mixture weights are functions of exogenous inputs, such as a detected bump or a proposed clinical action (Shao et al., 2020). The second, and the central idea of the paper, is *branch roles*. In a Defeasible Probabilistic Circuit (DPC), the branches of a gated node represent a default and its exceptions; in a Deontic Probabilistic Circuit (DEPC), they represent compliant and violating modes of behavior. The roles do more than aid interpretation: each one supports an exact inference operation (Fig. 1). The compliance role lets us ask what outcomes to expect if the protocol were followed, simply by rerouting the circuit, without training a second model. The priority attached to an exception lets us add a new rule to a trained model and bound how much any of its predictions can change as a result.

Our paper makes the following contributions: **1.** We introduce DPCs and DEPCs, extending conditional PCs with update gates and semantic branch roles, and prove that descriptive and normative queries remain exact and linear-time (Section 2); we also show that norms cannot be recovered from behavioral data and give a tractable construction for obligations triggered by violations. **2.** We show that any learned softmax gate compiles, without altering a prediction, into an ordered set of rules, where inserting an exception leaves every higher-priority rule untouched and a certificate from one forward and one backward pass bounds how much any conditional prediction can move (Section 3). **3.** We show how a DEPC audits behavior against a protocol, computing exactly how much more harm is expected than under the protocol and pointing to the parts of the model where behavior departs from it, both in time linear in circuit size (Section 4). **4.** Trained only on observations and never on rankings, DPCs and DEPCs rank the effects of updates and actions better than a capacity-matched conditional circuit, even on simulators with no circuit structure, and far better than three zero-shot language models with up to 32B parameters (Section 5).

2 Probabilistic Circuits with Semantic Branch Roles

We first introduce conditional PCs, then add the branch roles that define DPCs and DEPCs.

Notation: Random variables and their values are denoted by uppercase and lowercase letters respectively. Let $\mathbf{X} = \{X_1, \ldots, X_d\}$ denote a set of random variables and $\mathbf{x} = (x_1, \ldots, x_d)$ the joint assignment. For $\mathbf{S} \subseteq \mathbf{X}$, $\mathbf{x_S}$ restricts $\mathbf{x}$ to $\mathbf{S}$. Evidence is an assignment $\mathbf{e}$ to variables $\mathbf{E} \subseteq \mathbf{X}$, and a query $\mathbf{f}$ is an assignment to variables $\mathbf{F} \subseteq \mathbf{X}$. Exogenous variables $\mathbf{U}$ represent updates, actions, or external context; $\mathbf{u}$ denotes a particular setting and $\mathbf{u}_0$ a designated neutral setting.

A probabilistic circuit (PC) is a rooted DAG of input, product, and sum nodes (Choi et al., 2020). We use its conditional form, in which the weights of sum nodes may depend on an observed exogenous input $\mathbf{u}$ (Shao et al., 2020). For a circuit node n, $\mathrm{ch}(n)$ denotes its children and $\mathrm{sc}(n) \subseteq \mathbf{X}$ its scope, the variables represented below n. The root has scope $\mathbf{X}$. We write P_n for the distribution represented at node n, ϕ_n for the distribution at an input node, and w_{nc} for the weight on edge (n, c) from a sum node n to child c. In a conditional PC, these weights depend on $\mathbf{u}$ and are denoted by $w_{nc}(\mathbf{u})$.

Definition 1 (Conditional PC)**.** A conditional PC over $\mathbf{X}$ given $\mathbf{U}$ is a DAG whose nodes compute

$$P_n(\mathbf{x}_{\mathrm{sc}(n)} \mid \mathbf{u}) = \begin{cases} \phi_n(\mathbf{x}_{\mathrm{sc}(n)}) & n \text{ is an input node,} \\ \displaystyle\prod_{c \in \mathrm{ch}(n)} P_c(\mathbf{x}_{\mathrm{sc}(c)} \mid \mathbf{u}) & n \text{ is a product node,} \\ \displaystyle\sum_{c \in \mathrm{ch}(n)} w_{nc}(\mathbf{u}) P_c(\mathbf{x}_{\mathrm{sc}(c)} \mid \mathbf{u}) & n \text{ is a sum node.} \end{cases} \tag{1}$$

Each ϕ_n is normalized, $w_{nc}(\mathbf{u}) \geq 0$, and $\sum_{c \in \mathrm{ch}(n)} w_{nc}(\mathbf{u}) = 1$. An ordinary PC is the special case in which the sum weights do not depend on $\mathbf{u}$.

We focus on PCs that are *smooth and decomposable*. Smoothness means that the children of a sum node have identical scopes; decomposability means that the children of a product node have pairwise disjoint scopes. These conditions imply that every node represents a normalized distribution and that $P(\mathbf{e} \mid \mathbf{u})$ can be computed by one upward pass, linear in the number of circuit edges, once the gates have been evaluated. Conditional probabilities $P(\mathbf{f} \mid \mathbf{e}, \mathbf{u})$ are ratios of two such passes. The weights of a sum node s are produced by a *gate*: a learned network maps $\mathbf{u}$ to a logit $z_{sc}(\mathbf{u})$ for each child c, and $w_{sc}(\mathbf{u}) = \exp z_{sc}(\mathbf{u}) / \sum_{j \in \mathrm{ch}(s)} \exp z_{sj}(\mathbf{u})$. Gates may share a learned representation of $\mathbf{u}$. Fixing $\mathbf{u}$ evaluates every gate and yields an ordinary PC that represents the distribution $P(\mathbf{x}|\mathbf{u})$.

PCs also admit a latent-trace interpretation (Peharz et al., 2017; Zhao et al., 2016). A *trace* T selects one child at each reached sum node and all children at each reached product node, and contains one input node per variable. Sampling a trace according to the sum weights and then sampling its input distributions defines a joint distribution over $(T, \mathbf{X})$ whose marginal on $\mathbf{X}$ is the circuit distribution; a PC is thus a mixture over traces, each weighted by the product of its selected sum weights. Given evidence, one upward and one backward pass compute the posterior probability that a node or edge occurs in a trace, commonly called its *circuit flow* (Choi et al., 2021; Dang et al., 2022). We use the joint distribution over traces and outcomes in Section 4, and circuit flows in Sections 3 and 4.

A conditional PC can therefore learn how an update or action changes the distribution. What it cannot determine from the distribution alone is what its branches *mean*. Permuting the children of a sum node together with their weights leaves the represented distribution unchanged. Thus nothing in the conditional distribution identifies one branch as a default and another as an exception, or one as compliant and another as violating. DPCs and DePCs add precisely these roles.

2.1 Defeasible and deontic probabilistic circuits

Let $\mathcal{S}$ denote the set of sum nodes. We assign semantic roles only to selected nodes; all remaining sum nodes remain ordinary probabilistic mixtures.

Definition 2 (DPC)**.** A *defeasible probabilistic circuit* (DPC) is a conditional PC with a neutral input $\mathbf{u}_0$ and a designated subset $\mathcal{D} \subseteq \mathcal{S}$ of defeasible sum nodes. At each $s \in \mathcal{D}$, one child is designated the default and the remaining children are exceptions carrying a supplied priority order, with the default acting as the lowest-priority fallback.

The priority order is not inferred from the learned weights, and the designation changes no probability. It specifies how later revisions may act on the model. In Section 3 we show that any learned softmax gate can be converted exactly into an ordered-rule representation, and that a new exception can then be inserted without changing higher-priority rules.

Definition 3 (DePC)**.** A *deontic probabilistic circuit* (DePC) is a conditional PC with a designated subset $\mathcal{A} \subseteq \mathcal{S}$ of audited sum nodes and a reference policy. At each $s \in \mathcal{A}$, a nonempty set $\mathcal{N}_s \subseteq \mathrm{ch}(s)$ is designated compliant; the remaining children are violating.

A DePC has two readings of the same circuit. The *descriptive* distribution uses the learned gates and represents what the model predicts will happen. The *normative reference* uses the same circuit but routes probability according to the supplied policy. A simple reference policy masks violating children and renormalizes the compliant ones, namely $\nu_{sc}(\mathbf{u}) \propto w_{sc}(\mathbf{u}) \mathbb{1}\{c \in \mathcal{N}_s\}$ for $s \in \mathcal{A}$, which is well defined whenever the compliant children retain positive weight; a singleton $\mathcal{N}_s$ gives a hard clamp. Unaudited gates are evaluated at a designated reference input $\mathbf{u}_{\mathrm{ref}}$, typically $\mathbf{u}_0$, so that the reference does not depend on the candidate action being audited.

Since this replacement changes only sum-node weights, leaving the graph, scopes, and input distributions intact, $Q_{\mathcal{N}}$ remains smooth and decomposable, and its queries have the same linear-time complexity as the descriptive model's. Audits therefore cost what predictions cost.

The compliance specification, however, is not determined by the descriptive distribution.

Proposition 4 (Normative non-identifiability)**.** *The normative reference is not identifiable from the descriptive distribution alone. In particular, there exist distinct compliance specifications $\mathcal{N}$ and $\mathcal{N}'$ that are compatible with the same descriptive distribution P but induce different normative references, $Q_{\mathcal{N}} \neq Q_{\mathcal{N}'}$. Consequently, no function of P alone can recover the normative reference for every compliance specification.*

For example, consider a sum node representing $P = .5P_1 + .5P_2,\ P_1 \neq P_2$. If only the first child is designated compliant, then $Q_{\mathcal{N}} = P_1$; if only the second is designated compliant, then $Q_{\mathcal{N}'} = P_2$. The descriptive circuit and distribution are identical in the two cases, but the normative references differ. Hence observations from P can identify how behavior is distributed, but cannot by themselves identify which behavior is compliant. In our nurse example, if ward data show that nurses skip a double-check 30% of the time, the same descriptive distribution is compatible with a policy that forbids skipping and one that permits it.

The proposition concerns identification rather than learning in general: norms can of course be acquired from normative supervision, such as labeled violations or an encoded protocol. What it rules out is inferring them from behavior alone, however much of it is observed. This is why a DePC takes the compliance specification as an input, supplied by a policy, protocol, rule base, or other source of normative information, rather than estimating it.

3 Adding Rules to a Trained Model

A DPC is useful when the model itself must change. Suppose a new failure mode or exception is discovered after training. Adding another softmax component changes the weights of every existing branch, including higher-priority exceptions, so a rule about a rare vibration can weaken a rule about loss of support. Defeasible revision should instead preserve stronger rules and change only what the new exception is allowed to override. Three questions must be answered to make this precise: whether a trained gate can be put in a form where priority is meaningful, what an insertion into that form does to existing rules, and how far the edit can move predictions elsewhere in the circuit.

Can a learned gate be represented as ordered rules? Let P_0 be the default at a defeasible sum node s, and let $P_1, \ldots, P_k$ be its exceptions in decreasing priority. Associate exception i with an applicability $a_i \in [0, 1]$ and set $w_i = a_i \prod_{j<i}(1 - a_j)$ and $w_0 = \prod_{j=1}^{k}(1 - a_j)$, so $P_s = \sum_{i=1}^{k} w_i P_i + w_0 P_0$. Thus exception i receives mass only when no higher-priority exception has been selected, and the default receives the remainder. With binary applicabilities this selects the highest-priority applicable exception, which is the behavior prioritized default reasoning asks for; intermediate values interpolate between exceptions probabilistically.

Ordering the exceptions in this way would be of little use if it constrained what the gate can represent, since we intend to apply it to an already trained model.

Proposition 5 (Exact compilation)**.** *Given normalized weights $(\bar{w}_1, \ldots, \bar{w}_k, \bar{w}_0)$ and any supplied ordering of the exceptions, let $r_i = \bar{w}_0 + \sum_{j=i}^{k} \bar{w}_j$ and set $a_i = \bar{w}_i / r_i$ when $r_i > 0$ and $a_i = 0$*

otherwise. These applicabilities reproduce the original weights exactly. For softmax logits, $a_i = \mathrm{sigmoid}(z_i - \mathrm{LSE}(z_0, z_{i+1}, \ldots, z_k))$*, and all applicabilities are obtained in* $O(k)$ *time.*

Compilation therefore changes no probability, requires no retraining, and applies to any ordering: the data constrain the weights, not their order. This is what separates the two things a DPC keeps apart. The supplied ordering says nothing about what the gate learned; it specifies how future revisions are allowed to act on it.

How can a new exception be inserted without changing stronger rules? Suppose a new exception Q_s is inserted immediately before position $t \in \{1, \ldots, k+1\}$ with applicability $b \in [0, 1]$. Rules before t have higher priority and should remain unchanged. Let $R = \prod_{j<t}(1 - a_j)$ be the mass left after those rules and $H_s = \sum_{i<t} w_i P_i$ their contribution. Define the normalized lower-priority suffix $T_t = \sum_{i=t}^{k}(a_i \prod_{j=t}^{i-1}(1 - a_j))P_i + (\prod_{j=t}^{k}(1 - a_j))P_0$; hence $P_s = H_s + RT_t$, and the insertion should draw only on RT_t.

Theorem 6 (Priority-preserving insertion)**.** *After inserting* Q_s*, the edited node is* $P_s^+ = H_s + R[(1 - b)T_t + bQ_s]$*. Every higher-priority weight is unchanged, every old weight below the insertion point is multiplied by* $1 - b$*, and the new rule receives mass* $\delta = bR$*. With* $K_s = H_s + (1 - b)RT_t$*,*

$$P_s = K_s + \delta T_t, \qquad P_s^+ = K_s + \delta Q_s, \qquad \mathrm{TV}(P_s^+, P_s) = \delta\ \mathrm{TV}(Q_s, T_t) \le \delta. \tag{2}$$

The theorem gives the priority order an operational meaning: an exception can take mass only from the rules it is permitted to override, and never from a stronger one. A softmax append of weight η gives $(1 - \eta)P_s + \eta Q_s$ instead, which rescales every existing branch. In Fig. 1c, inserting a weaker vibration rule with mass .20 below a support-loss exception of weight .60 gives $(.60, .20, .20)$, whereas a softmax append gives $(.48, .20, .32)$ and weakens the stronger rule. Equation (2) also identifies the part of the node that the edit leaves alone, K_s, which is what the bounds below formalize.

What can a local edit change? Preserving weights is not the same as preserving predictions, and it is predictions that a deployed model is judged on. The edit occurs at one node, but the root distribution depends on that node together with the rest of the circuit. Let $\mathbf{S} = \mathrm{sc}(s)$ be the scope of the edited node. Decomposability makes the root distribution affine in P_s: $P(\mathbf{x}) = A_s(\mathbf{x}_{\mathbf{S}^c})P_s(\mathbf{x}_{\mathbf{S}}) + B_s(\mathbf{x})$, where $A_s, B_s \ge 0$ and neither changes during the edit. For evidence $\mathbf{e}$, let $D_s(\mathbf{e}) = \partial P(\mathbf{e})/\partial P_s(\mathbf{e}_{\mathbf{S}})$, obtained by one backward pass. Then $P^+(\mathbf{e}) - P(\mathbf{e}) = \delta D_s(\mathbf{e})[Q_s(\mathbf{e}_{\mathbf{S}}) - T_t(\mathbf{e}_{\mathbf{S}})]$: the effect of the edit anywhere in the circuit is the mass it moves, scaled by how much the query depends on the edited node.

Corollary 7 (Preservation outside the edited scope)**.** *The insertion leaves the marginal distribution over* $\mathbf{S}^c$ *unchanged. In particular, if the query and evidence involve only variables outside* $\mathbf{S}$*, then* $P^+(\mathbf{f} \mid \mathbf{e}) = P(\mathbf{f} \mid \mathbf{e})$.

A new exception therefore cannot change beliefs about variables it does not mention, which is the scope guarantee an engineer adding a rule about one subsystem would want. It is not the whole story: when the evidence involves variables inside $\mathbf{S}$, conditioning can amplify a small edit, and predictions about variables outside $\mathbf{S}$ can move as well.

How much can conditional predictions change? From Eq. (2), write the old and revised root distributions as $P = K + \delta A_s T_t$ and $P^+ = K + \delta A_s Q_s$. Under evidence $\mathbf{e}$, the mass the edit removes, the mass it adds, and the mass it leaves untouched are $v_{\mathbf{e}} = \delta D_s(\mathbf{e})T_t(\mathbf{e}_{\mathbf{S}})$, $u_{\mathbf{e}} = \delta D_s(\mathbf{e})Q_s(\mathbf{e}_{\mathbf{S}})$, and $r_{\mathbf{e}} = P(\mathbf{e}) - v_{\mathbf{e}}$. Conditioning renormalizes by the evidence probability, so the relevant comparison is between the moved mass and the mass the evidence retains.

Theorem 8 (Conditional revision certificate)**.** *Let* $Z = P(\mathbf{e}) = r_{\mathbf{e}} + v_{\mathbf{e}} > 0$ *and* $Z^+ = P^+(\mathbf{e}) = r_{\mathbf{e}} + u_{\mathbf{e}} > 0$*. Then* $\mathrm{TV}(P^+(\cdot \mid \mathbf{e}), P(\cdot \mid \mathbf{e})) \le \beta_s(\mathbf{e})$*, where* $\beta_s(\mathbf{e}) = \max\{v_{\mathbf{e}}/Z, u_{\mathbf{e}}/Z^+\} = 1 - r_{\mathbf{e}}/\max\{Z, Z^+\}$*. The bound is sharp.*

The certificate is evidence-specific, and it has to be. An edit that moves .01 of unconditional mass can move a conditional prediction by .910 if the evidence selects a rare context that the new rule happens to cover, so an unconditional bound such as δ would badly understate the risk. Because the bound is on total variation, it holds for *every* query at once: $|P^+(\mathbf{f} \mid \mathbf{e}) - P(\mathbf{f} \mid \mathbf{e})| \le \beta_s(\mathbf{e})$ without enumerating the queries it protects. Computing it needs one upward pass, one backward pass, and an evaluation of Q_s and T_t, so an edit can be audited before it is applied.

This turns into guarantees on decisions rather than on probabilities alone. A threshold decision at τ is unchanged whenever $|P(\mathbf{f} \mid \mathbf{e}) - \tau| > \beta_s(\mathbf{e})$; two candidates retain their ordering when their original gap exceeds the sum of their certificates; and for a severity function of range L, expected severity moves by at most $L\beta_s(\mathbf{e})$, so the audit scores of Section 4 inherit the same protection. A DPC therefore specifies both which existing rules a new exception may override and which earlier predictions and decisions survive the revision.

4 Auditing Behavior Against a Norm

A DePC gives two distributions on the same circuit: the descriptive distribution P, which represents what the model predicts will happen, and the normative reference Q, which represents what would happen under the supplied policy. An audit should answer two questions: *how much* does behavior depart from the policy, and *where* in the circuit does that departure occur? A distributional distance such as total variation answers neither well: it treats all disagreements alike, whereas an audit is concerned with shifts toward harmful outcomes, and it is not tractable to compute exactly between two circuits (Vergari et al., 2021). We therefore develop a severity-weighted gap and a decomposition of the discrepancy across gates that bounds this gap and localizes it. Fix an action or update $\mathbf{u}$ and evidence $\mathbf{e}$, and for brevity write $P = P(\cdot \mid \mathbf{u})$ and $Q = Q_{\mathcal{N}}(\cdot \mid \mathbf{u}_{\text{ref}})$.

4.1 Measuring how much behavior departs from the policy. Let $\sigma(\mathbf{x}) = \sum_i \sigma_i(x_i)$ be an *additive severity function*, with $\sigma_i \equiv 0$ for variables that carry no severity and larger values representing worse outcomes. For example, if X_i records an adverse-event level, one may set $\sigma_i(\text{none}) = 0$, $\sigma_i(\text{mild}) = 1$, $\sigma_i(\text{severe}) = 4$. We define the *audit gap* $G(\mathbf{e}, \mathbf{u}) = \mathbb{E}_P[\sigma \mid \mathbf{e}] - \mathbb{E}_Q[\sigma \mid \mathbf{e}]$, so that $G > 0$ means that behavior under the descriptive model has greater expected severity than behavior under the policy. The gap is useful only if it can be evaluated for the conditional queries an auditor poses, whose outcome spaces are too large to enumerate.

Lemma 9 (Exact conditional severity)**.** *If input node n represents X_i, let $p_n = \sum_{x_i \models \mathbf{e}} \phi_n(x_i)$ and $m_n = \sum_{x_i \models \mathbf{e}} \sigma_i(x_i)\phi_n(x_i)$. At sum nodes, $p_s = \sum_c w_{sc}(\mathbf{u}) p_c$ and $m_s = \sum_c w_{sc}(\mathbf{u}) m_c$; at product nodes, $p_s = \prod_c p_c$ and $m_s = \sum_c m_c \prod_{j \neq c} p_j$. Then $\mathbb{E}[\sigma \mid \mathbf{e}] = m_{\text{root}}/p_{\text{root}}$.*

The pair (p_n, m_n) is the evidence probability of the sub-circuit below n and its unnormalized severity moment. At a product node, a child's severity is realized only on the mass consistent with the evidence at its siblings, hence the weighting by their probabilities. The computation is one upward pass (Vergari et al., 2021), so an audit costs no more than the prediction it audits.

Proposition 10 (Properties of the audit gap)**.** *Let $L = \max_{\mathbf{x}} \sigma(\mathbf{x}) - \min_{\mathbf{x}} \sigma(\mathbf{x})$. Then $G = \sum_i G_i$, where G_i is the gap induced by σ_i; $|G| \leq L \ \text{TV}(P(\cdot \mid \mathbf{e}), Q(\cdot \mid \mathbf{e})) \leq L\sqrt{\text{KL}(Q(\cdot \mid \mathbf{e}) \| P(\cdot \mid \mathbf{e}))/2}$; and replacing σ by $a\sigma + b$, $a > 0$, multiplies G by a.*

Each part serves a purpose. The decomposition attributes a gap to individual outcome variables, which is what an auditor reports. Affine invariance means the origin and unit of the severity scale do not affect rankings, so only the relative weighting of outcomes matters, and that is a judgment the auditor supplies. The inequality links the two halves of this section: any bound on the distributional discrepancy bounds the severity gap simultaneously for every severity function of range at most L.

Example 11 (A worked audit)**.** Suppose an outcome has severities $(0, 1, 4)$ for none, mild, and severe harm. A compliant branch has distribution $(.90, .10, 0)$ and a violating branch $(.10, .30, .60)$. If an action assigns weights $(.7, .3)$, the descriptive distribution has expected severity .88, while the normative reference keeps only the compliant branch, with severity .10. The audit gap is therefore .78, of which .72 comes from severe outcomes: most mass remains on "no harm," so the distributions appear unremarkable, while the decomposition locates the gap in the outcome that matters.

4.2 Locating where behavior departs from the policy. Since P and Q share their graph and input distributions and differ only in how sum nodes route probability, the gates are where a departure can be located. A gate should contribute when routing there differs between model and policy *and* the gate is relevant under the evidence; a large difference at a gate this evidence never reaches is not a finding. For $M \in \{P, Q\}$, let $M_s(\mathbf{e})$ be the evidence probability at node s, let $\widehat{w}^M_{sc}(\mathbf{e}) = w^M_{sc} M_c(\mathbf{e})/M_s(\mathbf{e})$ be the posterior probability of taking child c given that a trace reaches s, and let $\rho^M_s(\mathbf{e}) = [M_s(\mathbf{e})/M(\mathbf{e})][\partial M(\mathbf{e})/\partial M_s(\mathbf{e})]$ be the posterior probability that a trace reaches s;

both are circuit flows obtained from one upward and one backward pass. Posterior rather than prior weights are needed here, since two gates can carry identical weights in P and Q yet differ in posterior when the evidence is explained by different descendants. Combining relevance with local difference gives the *gate-wise audit budget* $B_{\mathbf{e}} = \sum_{s \in \mathcal{S}} \rho_s^Q(\mathbf{e})\ \mathrm{KL}(\widehat{w}_s^Q(\mathbf{e}) \,\|\, \widehat{w}_s^P(\mathbf{e}))$.

Theorem 12 (Gate-wise audit bound)**.** $\mathrm{KL}(Q(\cdot \mid \mathbf{e}) \| P(\cdot \mid \mathbf{e})) \le B_{\mathbf{e}}$ *and* $|G(\mathbf{e}, \mathbf{u})| \le L \min\{1, \sqrt{B_{\mathbf{e}}/2}\}$.

The theorem turns a plausible attribution into an audit. It is conservative: the local terms cannot sum to less than the true divergence, so no part of the departure goes unaccounted for. It bounds $|G|$ for every severity function of range L, so one evaluation certifies audits whose severity function is not yet fixed. And since each summand is nonnegative and attached to a named gate, the largest terms identify where to inspect, while $B_{\mathbf{e}} = 0$ certifies that the two conditionals agree, so a compliant model is recognized as compliant rather than merely undetected.

Proposition 13 (Trace interpretation)**.** *Let T denote the circuit trace. Whenever $B_{\mathbf{e}}$ is finite,* $B_{\mathbf{e}} = \mathrm{KL}(Q(T, \mathbf{X} \mid \mathbf{e}) \| P(T, \mathbf{X} \mid \mathbf{e})) = \mathrm{KL}(Q(\mathbf{X} \mid \mathbf{e}) \| P(\mathbf{X} \mid \mathbf{e})) + \mathbb{E}_Q[\mathrm{KL}(Q(T \mid \mathbf{X}, \mathbf{e}) \| P(T \mid \mathbf{X}, \mathbf{e}))]$.

The bound is therefore exact for the joint distribution over traces and outcomes, and its excess over the outcome divergence is the expected disagreement about which route produced an outcome. That excess is informative rather than slack: two policies can produce similar outcomes by different means, and an audit asking where practice departs from protocol needs that difference to remain visible. In Example 11, $B_{\mathbf{e}} = -\log .7 = .357$ nats, the outcome divergence is .232, and the remaining .125 is routing disagreement invisible in the outcomes.

4.3 Using the audit to rank actions and updates. Let $\mathbf{u}_0$ denote a neutral action. For a query assignment $\mathbf{f}$, define $S_{\mathbf{f}}(\mathbf{e}, \mathbf{u}) = P(\mathbf{f} \mid \mathbf{e}, \mathbf{u}) - P(\mathbf{f} \mid \mathbf{e}, \mathbf{u}_0)$ and $S_\sigma(\mathbf{e}, \mathbf{u}) = G(\mathbf{e}, \mathbf{u}) - G(\mathbf{e}, \mathbf{u}_0)$. The first measures how an update changes the query probability; the second measures how an action changes the audit gap relative to the neutral action, which removes the severity the context contributes on its own and would otherwise dominate comparisons in an already risky context.

If all candidates share the same normative reference, it cancels: $S_\sigma(\mathbf{e}, \mathbf{u}) = \mathbb{E}_{P(\cdot \mid \mathbf{e}, \mathbf{u})}[\sigma] - \mathbb{E}_{P(\cdot \mid \mathbf{e}, \mathbf{u}_0)}[\sigma]$. Our severity experiments therefore test whether the compliance/violation structure helps a DePC predict the consequences of different actions; the reference fixes the scale and the meaning of a single action's score rather than the ordering. If the reference depends on the candidate, it does not cancel and the full audit of Theorem 12 applies.

5 Experiments

We evaluate whether circuits trained only on observational data can recover rankings over candidate updates or actions without ranking supervision. They learn the underlying conditional distributions, from which rankings are derived at test time.

5.1 Experimental Setup

Datasets. The DPC physical benchmark contains 32 categorical variables and 109 updates, with 100,000 training and 10,000 validation observations. Each of 1,080 evaluation instances contains a partially observed context, an outcome hypothesis, and 8 candidate updates ranked by how strongly they strengthen or weaken the hypothesis. The DePC clinical benchmark contains 32 categorical world variables and 8 action variables covering 31 action configurations, with the same training and validation sizes. Each of 1,080 instances contains a clinical context and 8 candidate actions, such as verification, procedural overrides, or independent double-checks, ranked by relative severity.

Baselines. **CSPN** is our primary architectural baseline. Like DPC and DePC, its sum-node weights depend on the update, but its branches lack the proposed Default/Exception or Compliance/Violation semantics. **Vanilla PC** uses the same observations to model a static joint distribution. We also compare against the commercial **GPT-5.6 Sol**[1], **Qwen3.8-27B**[2], and **Qwen3-32B** (Yang et al., 2025).

[1] https://developers.openai.com/api/docs/models/gpt-5.6-sol

[2] https://huggingface.co/Qwen/Qwen3.8-27B

Table 1: Main physical and clinical ranking results. Accuracies are percentages; best results are bold.

	Model	Params	Strict τ	Tie τ	Pairwise %	Top-1 %	Exact %
Physical	DPC (ours)	0.936M	**0.5953**	**0.6165**	**80.90**	55.83	**1.20**
	CSPN	0.932M	0.4700	0.4908	74.61	**56.85**	0.19
	Vanilla PC	0.087M	0.3836	0.3982	69.98	50.28	0.09
	GPT-5.6 Sol	–	0.2869	0.2982	64.98	36.76	0.37
	Qwen3-32B	32B	0.2556	0.2653	63.32	35.00	0.00
	Qwen3.8-27B	27B	0.2899	0.3022	65.19	36.02	0.37
Clinical	DePC (ours)	0.298M	**0.6757**	**0.6960**	**84.75**	**50.09**	**3.43**
	CSPN	0.296M	0.6645	0.6841	84.16	44.07	2.59
	Vanilla PC	0.276M	0.5050	0.5191	75.90	40.93	0.37
	GPT-5.6 Sol	–	0.4921	0.5038	75.17	40.28	0.83
	Qwen3-32B	32B	0.4534	0.4648	73.23	34.17	0.46
	Qwen3.8-27B	27B	0.4637	0.4741	73.71	37.69	0.28

Learning. The descriptive model is learned from observations by maximizing conditional likelihood, $\sum_n \log P_\theta(\mathbf{x}^{(n)} \mid \mathbf{u}^{(n)})$, with standard gradient or EM updates for conditional PCs; no ranking labels are used and checkpoints are chosen by validation likelihood. Semantic information is supplied separately: compliance labels and normative weights come from the policy, and DPC priorities specify how later revisions may act. The separation is forced by Proposition 4 and is also deliberate, since fitting P to reduce the audit gap would pull the descriptive model toward the policy and erase the discrepancy the audit is meant to detect. The revision guarantees extend to learning as well: a new exception can be fitted on fresh data with the rest of the circuit frozen, in which case Theorem 6 prevents any change to higher-priority rules and constraining $\beta_s(\mathbf{e})$ on a set of protected contexts bounds, by Theorem 8, how far any earlier conditional prediction can move.

Metrics. We report strict Kendall's τ, tie-aware Kendall's τ, pairwise accuracy, top-1 accuracy, and exact ranking accuracy. Strict Kendall's τ compares complete predicted and ground-truth rankings; pairwise accuracy measures the fraction of correctly ordered candidate pairs; top-1 accuracy checks whether the highest-ranked candidate matches the ground truth; and exact ranking accuracy requires the full 8-candidate ordering to match the ground-truth ranking. Tie-aware Kendall's τ avoids imposing an arbitrary strict ordering when candidate scores are nearly indistinguishable.

5.2 Results

Table 1 shows that DPC improves physical ranking over CSPN: strict Kendall's τ increases from 0.4700 to 0.5953, tie-aware τ from 0.4908 to 0.6165, pairwise accuracy from 74.61% to 80.90%, and exact-ranking accuracy from 0.19% to 1.20%. CSPN is slightly higher on top-1 accuracy, at 56.85% versus 55.83% for DPC. On the clinical domain, DePC yields smaller gains: strict τ increases from 0.6645 to 0.6757, tie-aware τ from 0.6841 to 0.6960, pairwise accuracy from 84.16% to 84.75%, top-1 accuracy from 44.07% to 50.09%, and exact-ranking accuracy from 2.59% to 3.43%. Paired bootstrap differences are $+0.0112$ for strict τ, with a 95% confidence interval (CI) of $[0.0017, 0.0206]$, and $+0.0059$ for pairwise accuracy, with a 95% CI of $[0.0013, 0.0106]$. Vanilla PC and all three LLM baselines perform below the proposed models on the main ranking metrics.

Generalization Evaluation. We further evaluate the models on two independently constructed generalization benchmarks for the physical and clinical settings. Each benchmark is generated by a separately specified stochastic simulator rather than the generator used for the corresponding main benchmark. For each benchmark, we generate a new learning corpus, retrain all circuit models from scratch, and evaluate them on the corresponding ranking test set. Table 2 shows that the improvements persist on the independently generated datasets. On the physical domain, DPC improves strict Kendall's τ from 0.6806 to 0.7272, tie-aware τ from 0.7283 to 0.7758, and pairwise accuracy from 86.42% to 88.83% over CSPN. It also improves top-1 accuracy from 76.94% to 79.54% and exact-ranking accuracy from 2.59% to 3.24%. On the clinical domain, DePC improves strict Kendall's τ from 0.7685 to 0.7792, tie-aware τ from 0.8508 to 0.8617, and pairwise accuracy from 92.46% to 92.99%. Top-1 accuracy increases slightly from 74.07% to 74.35%, while exact-ranking accuracy

Table 2: Ranking results on independently constructed generalization benchmarks for the physical and clinical settings. All circuit models are retrained on the corresponding learning corpus.

	Model	Strict τ	Tie τ	Pairwise %	Top-1 %	Exact %
Physical	DPC (ours)	**0.7272**	**0.7758**	**88.83**	**79.54**	**3.24**
	CSPN	0.6806	0.7283	86.42	76.94	2.59
	Vanilla PC	0.6760	0.7215	86.07	75.37	0.37
Clinical	DePC (ours)	**0.7792**	**0.8617**	**92.99**	**74.35**	**4.63**
	CSPN	0.7685	0.8508	92.46	74.07	4.07
	Vanilla PC	0.6390	0.7055	85.10	38.61	1.02

increases from 4.07% to 4.63%. Overall, the same performance pattern is observed on independently generated datasets, suggesting that the gains are not restricted to benchmark construction.

6 Related Work

Probabilistic circuits. Sum-product networks (Poon & Domingos, 2011; Gens & Domingos, 2013; Peharz et al., 2015), cutset networks (Rahman et al., 2014), and PSDDs (Kisa et al., 2014) are instances of probabilistic circuits (Darwiche & Marquis, 2002; Choi et al., 2020; Vergari et al., 2021). Conditional SPNs (Shao et al., 2020) and conditional cutset networks (Rahman et al., 2019), our closest architectural relative and main baseline, make sum weights functions of an external input; mixtures of experts use a similar gating mechanism (Jordan & Jacobs, 1994). These models, however, treat sum-node branches as semantically interchangeable. DPCs and DePCs assign roles to branches at selected gates. The DPC parameterization is related to finite stick-breaking (Ren et al., 2011), but we use it to obtain exact compilation, priority-preserving edits, and conditional-prediction guarantees.

Defeasible and Deontic reasoning. Default logics, argumentation, preferential entailment, and prioritized defaults provide symbolic accounts of exceptions (Reiter, 1980; Nute, 2003; Pollock, 1987; Dung, 1995; Kraus et al., 1990; Lehmann & Magidor, 1992; Grosof, 1991). DPCs place a supplied ordering of defaults and exceptions inside a learned probabilistic circuit. AGM revision (Alchourrón et al., 1985) and the update–revision distinction (Katsuno & Mendelzon, 1991) motivate our separation between changing the input to a fixed model and structurally adding a rule. Deontic logic separates what occurs from what is obligatory or permitted (von Wright, 1951), while contrary-to-duty reasoning studies obligations triggered by earlier violations (Chisholm, 1963; Prakken & Sergot, 1996; Carmo & Jones, 2002; Makinson & Van Der Torre, 2000). DePCs do not propose a new deontic logic. They use a supplied policy to construct a normative reference, quantify severity-weighted departures from it, and identify where the descriptive and normative distributions differ.

Neuro-symbolic learning and model editing. Neuro-symbolic methods incorporate logical structure into learning (Manhaeve et al., 2018; Xu et al., 2018; Raedt et al., 2020). MEND and ROME (Mitchell et al., 2022; Meng et al., 2022) evaluate model-editing side effects empirically. Our circuit analysis instead provides an evidence-specific certificate that bounds the effect of an edit on every query.

7 Limitations and Conclusion

All four evaluation environments are synthetic. This permits controlled comparisons, including one benchmark generated by an independent simulator with no circuit structure, but the clinical environments remain testbeds rather than evidence of clinical utility. The revision and audit guarantees are exact under their stated assumptions; learned edits after distribution shift are not evaluated. The severity experiments use a reference shared across candidates, and the reported intervals capture variation across queries rather than training runs. Priorities, compliance policies, and severity scales are external inputs; Proposition 4 shows that norms cannot generally be recovered from observed behavior alone. Within this scope, DPCs support certified, priority-preserving revision, while DePCs compare learned behavior with a supplied policy and localize where they differ. Without ranking supervision, they improve update and action ranking, respectively, over comparably sized conditional circuits and three zero-shot language models.

AI USE STATEMENT

We used generative AI tools (Claude and ChatGPT) to help draft and edit the text, develop and check proofs, write the programs that verify the mathematical identities numerically, and find related work. We did not use them to generate data, run the benchmark experiments, or produce the reported results. We reviewed all AI-assisted text, proofs and code, and we take full responsibility for the content of this paper.

ETHICS STATEMENT

The clinical domain is synthetic and contains no patient data. Its severity weights and compliance labels were chosen by us to test probabilistic reasoning; an audit computed by our method is only as good as the policy and utilities it is given, and any deployment would need domain experts to set and review both.

REFERENCES

Carlos E. Alchourrón, Peter Gärdenfors, and David Makinson. On the logic of theory change: Partial meet contraction and revision functions. *J. Symb. Log.*, 50(2):510–530, 1985. doi: 10.2307/2274239. URL https://doi.org/10.2307/2274239.

José Carmo and Andrew JI Jones. Deontic logic and contrary-to-duties. In *Handbook of Philosophical Logic: Volume 8*, pp. 265–343. Springer, 2002.

Roderick M Chisholm. Contrary-to-duty imperatives and deontic logic. *Analysis*, 24(2):33–36, 1963.

YooJung Choi, Antonio Vergari, and Guy Van den Broeck. Probabilistic circuits: A unifying framework for tractable probabilistic models. Technical report, University of California, Los Angeles, 2020. URL http://starai.cs.ucla.edu/papers/ProbCirc20.pdf.

YooJung Choi, Meihua Dang, and Guy Van den Broeck. Group fairness by probabilistic modeling with latent fair decisions. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pp. 12051–12059. AAAI Press, 2021. doi: 10.1609/AAAI.V35I13.17431. URL https://doi.org/10.1609/aaai.v35i13.17431.

Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory*. John Wiley & Sons, 2 edition, 2006. ISBN 9780471241959. doi: 10.1002/047174882X. URL https://doi.org/10.1002/047174882X.

Meihua Dang, Antonio Vergari, and Guy Van den Broeck. Strudel: A fast and accurate learner of structured-decomposable probabilistic circuits. *International Journal of Approximate Reasoning*, 140:92–115, 2022. ISSN 0888-613X. doi: https://doi.org/10.1016/j.ijar.2021.09.012. URL https://www.sciencedirect.com/science/article/pii/S0888613X21001547.

Adnan Darwiche and Pierre Marquis. A knowledge compilation map. *J. Artif. Int. Res.*, 17(1): 229–264, September 2002. ISSN 1076-9757.

Phan Minh Dung. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *Artificial intelligence*, 77(2):321–357, 1995.

Robert Gens and Pedro Domingos. Learning the structure of sum-product networks. In Sanjoy Dasgupta and David McAllester (eds.), *Proceedings of the 30th International Conference on Machine Learning*, volume 28 of *Proceedings of Machine Learning Research*, pp. 873–880, Atlanta, Georgia, USA, 17–19 Jun 2013. PMLR. URL https://proceedings.mlr.press/v28/gens13.html.

Benjamin N. Grosof. Generalizing prioritization. In James F. Allen, Richard Fikes, and Erik Sandewall (eds.), *Proceedings of the 2nd International Conference on Principles of Knowledge Representation and Reasoning (KR'91). Cambridge, MA, USA, April 22-25, 1991*, pp. 289–300. Morgan Kaufmann, 1991.

Michael I. Jordan and Robert A. Jacobs. Hierarchical mixtures of experts and the em algorithm. *Neural Computation*, 6(2):181–214, 1994. doi: 10.1162/neco.1994.6.2.181. URL https://doi.org/10.1162/neco.1994.6.2.181.

Hirofumi Katsuno and Alberto O. Mendelzon. On the difference between updating a knowledge base and revising it. In James F. Allen, Richard Fikes, and Erik Sandewall (eds.), *Proceedings of the 2nd International Conference on Principles of Knowledge Representation and Reasoning (KR'91). Cambridge, MA, USA, April 22-25, 1991*, pp. 387–394. Morgan Kaufmann, 1991.

Doga Kisa, Guy Van den Broeck, Arthur Choi, and Adnan Darwiche. Probabilistic sentential decision diagrams. In *Principles of Knowledge Representation and Reasoning: Proceedings of the Fourteenth International Conference, KR 2014*, pp. 558–567. AAAI Press, 2014. URL https://aaai.org/papers/0558-8005-probabilistic-sentential-decision-diagrams/.

Sarit Kraus, Daniel Lehmann, and Menachem Magidor. Nonmonotonic reasoning, preferential models and cumulative logics. *Artif. Intell.*, 44(1–2):167–207, July 1990. ISSN 0004-3702. doi: 10.1016/0004-3702(90)90101-5. URL https://doi.org/10.1016/0004-3702(90)90101-5.

Daniel Lehmann and Menachem Magidor. What does a conditional knowledge base entail? *Artif. Intell.*, 55(1):1–60, 1992. doi: 10.1016/0004-3702(92)90041-U. URL https://doi.org/10.1016/0004-3702(92)90041-U.

David Makinson and Leendert Van Der Torre. Input/output logics. *Journal of philosophical logic*, 29 (4):383–408, 2000.

Robin Manhaeve, Sebastijan Dumancic, Angelika Kimmig, Thomas Demeester, and Luc De Raedt. Deepproblog: Neural probabilistic logic programming. In Samy Bengio, Hanna M. Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett (eds.), *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pp. 3753–3763, 2018. URL https://proceedings.neurips.cc/paper/2018/hash/dc5d637ed5e62c36ecb73b654b05ba2a-Abstract.html.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. Locating and editing factual associations in GPT. In Sanmi Koyejo, S. Mohamed, A. Agarwal, Danielle Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022. URL http://papers.nips.cc/paper_files/paper/2022/hash/6f1d43d5a82a37e89b0665b33bf3a182-Abstract-Conference.html.

Eric Mitchell, Charles Lin, Antoine Bosselut, Chelsea Finn, and Christopher D. Manning. Fast model editing at scale. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022. URL https://openreview.net/forum?id=0DcZxeWfOPt.

Donald Nute. Defeasible logic. In Oskar Bartenstein, Ulrich Geske, Markus Hannebauer, and Osamu Yoshie (eds.), *Web Knowledge Management and Decision Support*, pp. 151–169, Berlin, Heidelberg, 2003. Springer Berlin Heidelberg. ISBN 978-3-540-36524-2.

Robert Peharz, Sebastian Tschiatschek, Franz Pernkopf, and Pedro Domingos. On Theoretical Properties of Sum-Product Networks. In Guy Lebanon and S. V. N. Vishwanathan (eds.), *Proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics*, volume 38 of *Proceedings of Machine Learning Research*, pp. 744–752, San Diego, California, USA, 09–12 May 2015. PMLR. URL https://proceedings.mlr.press/v38/peharz15.html.

Robert Peharz, Robert Gens, Franz Pernkopf, and Pedro Domingos. On the latent variable interpretation in sum-product networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(10):2030–2044, 2017. doi: 10.1109/TPAMI.2016.2618381.

John L Pollock. Defeasible reasoning. *Cognitive science*, 11(4):481–518, 1987.

Hoifung Poon and Pedro Domingos. Sum-product networks: A new deep architecture. In *Proceedings of the Twenty-Seventh Conference on Uncertainty in Artificial Intelligence (UAI 2011)*, pp. 337–346. AUAI Press, 2011.

Henry Prakken and Marek Sergot. Contrary-to-duty obligations. *Studia Logica*, 57(1):91–115, 1996.

Luc de Raedt, Sebastijan Dumančić, Robin Manhaeve, and Giuseppe Marra. From statistical relational to neuro-symbolic artificial intelligence. In Christian Bessiere (ed.), *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, pp. 4943–4950. International Joint Conferences on Artificial Intelligence Organization, 7 2020. doi: 10.24963/ijcai.2020/688. URL https://doi.org/10.24963/ijcai.2020/688. Survey track.

Tahrima Rahman, Prasanna V. Kothalkar, and Vibhav Gogate. Cutset networks: A simple, tractable, and scalable approach for improving the accuracy of chow-liu trees. In *Machine Learning and Knowledge Discovery in Databases – European Conference, ECML PKDD 2014, Nancy, France, September 15–19, 2014, Proceedings, Part II*, pp. 630–645. Springer, 2014. doi: 10.1007/978-3-662-44851-9_40.

Tahrima Rahman, Shasha Jin, and Vibhav Gogate. Cutset bayesian networks: A new representation for learning rao-blackwellised graphical models. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pp. 5751–5757. International Joint Conferences on Artificial Intelligence Organization, 7 2019. doi: 10.24963/ijcai.2019/797. URL https://doi.org/10.24963/ijcai.2019/797.

Raymond Reiter. A logic for default reasoning. *Artif. Intell.*, 13(1-2):81–132, 1980. doi: 10.1016/0004-3702(80)90014-4. URL https://doi.org/10.1016/0004-3702(80)90014-4.

Lu Ren, Lan Du, Lawrence Carin, and David Dunson. Logistic stick-breaking process. *Journal of Machine Learning Research*, 12(7):203–239, 2011. URL http://jmlr.org/papers/v12/ren11a.html.

Xiaoting Shao, Alejandro Molina, Antonio Vergari, Karl Stelzner, Robert Peharz, Thomas Liebig, and Kristian Kersting. Conditional sum-product networks: Imposing structure on deep probabilistic architectures. In Manfred Jaeger and Thomas Dyhre Nielsen (eds.), *International Conference on Probabilistic Graphical Models, PGM 2020, 23-25 September 2020, Aalborg, Hotel Comwell Rebild Bakker, Skørping, Denmark*, volume 138 of *Proceedings of Machine Learning Research*, pp. 401–412. PMLR, 2020. URL http://proceedings.mlr.press/v138/shao20a.html.

Antonio Vergari, YooJung Choi, Anji Liu, Stefano Teso, and Guy Van den Broeck. A compositional atlas of tractable circuit operations for probabilistic inference. In *Advances in Neural Information Processing Systems*, volume 34, pp. 13189–13201. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper/2021/hash/6e01383fd96a17ae51cc3e15447e7533-Abstract.html.

G. H. von Wright. Deontic logic. *Mind*, 60(237):1–15, 1951. ISSN 00264423, 14602113. URL http://www.jstor.org/stable/2251395.

Jingyi Xu, Zilu Zhang, Tal Friedman, Yitao Liang, and Guy Van den Broeck. A semantic loss function for deep learning with symbolic knowledge. In Jennifer Dy and Andreas Krause (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 5502–5511. PMLR, 10–15 Jul 2018. URL https://proceedings.mlr.press/v80/xu18h.html.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report, 2025. URL https://arxiv.org/abs/2505.09388.

Han Zhao, Pascal Poupart, and Geoffrey J. Gordon. A unified approach for learning the parameters of sum-product networks. In Daniel D. Lee, Masashi Sugiyama, Ulrike von Luxburg, Isabelle Guyon, and Roman Garnett (eds.), *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pp. 433–441, 2016. URL https://proceedings.neurips.cc/paper/2016/hash/6c9882bbac1c7093bd25041881277658-Abstract.html.

A PROOFS

We use the notation of Section 2. All variables have finite domains, all circuits are smooth and decomposable, and every node has nonempty scope. Input distributions ϕ_n are normalized and admit tractable evidence evaluation. We fix the exogenous input $\mathbf{u}$ and suppress it unless different inputs are compared. For a node s, let $\mathbf{S} = \mathrm{sc}(s)$ and $\mathbf{S}^c = \mathbf{X} \setminus \mathbf{S}$. The notation $\mathbf{e}_\mathbf{S}$ denotes the part of the evidence assigning variables in $\mathbf{S}$, and $M_s(\mathbf{e})$ abbreviates $M_s(\mathbf{e}_{\mathrm{sc}(s)})$. Every conditional distribution below requires positive evidence probability under the corresponding model. Complexity statements count circuit operations after gate and input evaluation. All logarithms are natural.

A.1 NORMALIZATION, REFERENCE POLICIES, AND TRACES

Normalization and exact inference. Fix all gate weights. An input is normalized by assumption. At a smooth sum node, the children are normalized distributions on the same scope, and their nonnegative weights sum to one. Their mixture is therefore normalized. At a decomposable product node, the child scopes are disjoint, so summing the product over its scope factors into a product of sums, each equal to one. Induction over the DAG proves normalization at every node, including the root.

A reference policy replaces gate weights by other normalized weights without changing the graph or the input distributions. The same induction therefore applies to the normative reference. For the masking policy in Section 2, the sum of the descriptive weights on compliant children must be positive. A supplied normalized policy need not be obtained by masking, but it must have the specified support at audited nodes.

For evidence $\mathbf{e}$, evaluate each input on the restriction of the evidence to its scope and propagate sums and products once. Smoothness and decomposability make this the exact marginal $M(\mathbf{e})$. For compatible assignments $\mathbf{e}$ and $\mathbf{f}$,

$$M(\mathbf{f} \mid \mathbf{e}) = \frac{M(\mathbf{e} \wedge \mathbf{f})}{M(\mathbf{e})}.$$

Here $\mathbf{e} \wedge \mathbf{f}$ is their joint assignment; if they conflict, the conditional probability is zero. This proves the inference claims in Section 2 for both descriptive and reference circuits.

Proof of Proposition 4. Let a root sum represent $P = \frac{1}{2}P_1 + \frac{1}{2}P_2$, with $P_1 \neq P_2$. If only the first child is compliant, masking gives reference P_1; if only the second child is compliant, it gives reference P_2. Both specifications have the same graph, input distributions, gate weights, and descriptive distribution. Consequently, they induce the same law for every sample drawn from P, but have different normative references. A function of P alone has the same value in both cases and cannot return the correct reference for both. □

Traces in a shared DAG. Unfold the circuit into a tree while retaining each node's original identity. A trace selects one child at each reached sum and all children at each reached product. A trace cannot contain two copies of the same node s. The copies cannot be ancestors of each other because the circuit is acyclic. Their lowest common ancestor in the unfolded trace would therefore have to be a product node. Its two relevant children would both contain the nonempty scope of s, contradicting decomposability. This argument also applies to nodes shared by several parents.

Let $\mathcal{S}(t)$ and $\mathcal{L}(t)$ be the sum nodes and input nodes in a trace t, respectively, and let $t(s)$ be the child selected at s. For a circuit M, the joint distribution over traces and assignments is

$$M(t, \mathbf{x}) = \prod_{s \in \mathcal{S}(t)} w^M_{s,t(s)} \prod_{\ell \in \mathcal{L}(t)} \phi_\ell(\mathbf{x}_{\mathrm{sc}(\ell)}). \tag{3}$$

The input scopes in a trace partition $\mathbf{X}$. With univariate inputs, there is one input per variable, as in Section 2. Summing Eq. (3) over traces gives the circuit distribution.

For $M_s(\mathbf{e}) > 0$, the posterior child weights and the posterior probability of reaching s are

$$\widehat{w}^M_{sc}(\mathbf{e}) = \frac{w^M_{sc} M_c(\mathbf{e})}{M_s(\mathbf{e})}, \qquad \rho^M_s(\mathbf{e}) = \frac{M_s(\mathbf{e})}{M(\mathbf{e})} \frac{\partial M(\mathbf{e})}{\partial M_s(\mathbf{e})}. \tag{4}$$

The derivative is with respect to the node output. A backward pass sums contributions from every parent, so the formula includes every occurrence of a shared node in the circuit computation. The child weights are conditional probabilities given that s is reached; the edge flow is $\rho^M_s(\mathbf{e})\widehat{w}^M_{sc}(\mathbf{e})$.

For a trace with positive evidence mass,

$$M(t \mid \mathbf{e}) = \prod_{s \in \mathcal{S}(t)} \widehat{w}^M_{s,t(s)}(\mathbf{e}). \tag{5}$$

To verify this identity, substitute Eq. (4). At each product node, its evidence value is the product of its children's values. All intermediate ratios cancel, leaving the product of the selected weights and the input evidence probabilities, divided by $M(\mathbf{e})$. Nodes with zero posterior flow are never visited and need no posterior child distribution.

A.2 Rule Compilation and Local Revision

Conditions on an edit. The results in Section 3 concern an edit at one designated defeasible node. The new component Q_s is normalized on $\mathbf{S} = \mathrm{sc}(s)$ and is represented by a smooth, decomposable subcircuit attached without creating a cycle. All existing component distributions, applicabilities, and parameters outside the stated local edit are fixed. In particular, an edit does not retrain a shared gate encoder. These conditions preserve smoothness and decomposability and ensure that the rest of the circuit remains fixed.

Proof of Proposition 5. Use the residual masses from the proposition, $r_i = \bar{w}_0 + \sum_{j=i}^{k} \bar{w}_j$, and set $r_{k+1} = \bar{w}_0$. Since $r_1 = 1$ and $r_{i+1} = (1 - a_i)r_i$, induction gives

$$r_i = \prod_{j<i}(1 - a_j), \qquad a_i \prod_{j<i}(1 - a_j) = \bar{w}_i.$$

The default receives $r_{k+1} = \bar{w}_0$. If $r_i = 0$, nonnegativity implies that every weight remaining in the suffix is zero. The prefix product has already vanished, so setting all subsequent applicabilities to zero reproduces those weights as well.

For finite softmax logits, canceling the common normalizer gives

$$a_i = \frac{e^{z_i}}{e^{z_i} + e^{z_0} + \sum_{j>i} e^{z_j}} = \mathrm{sigmoid}\big(z_i - \mathrm{LSE}(z_0, z_{i+1}, \ldots, z_k)\big).$$

Initialize $\ell_{k+1} = z_0$ and compute $\ell_i = \mathrm{LSE}(z_i, \ell_{i+1})$ in reverse order. Then $a_i = \mathrm{sigmoid}(z_i - \ell_{i+1})$, so all applicabilities take $O(k)$ time after logit evaluation. Compilation preserves every weight, hence every circuit prediction. A different order preserves the distribution when its applicabilities are recompiled; reordering at fixed applicabilities is a separate edit. □

Proof of Theorem 6. Let R and H_s be as in the main text. The normalized suffix is

$$T_t = \sum_{i=t}^{k} \left(a_i \prod_{j=t}^{i-1}(1 - a_j) \right) P_i + \left(\prod_{j=t}^{k}(1 - a_j) \right) P_0.$$

Empty products are one and empty sums are zero. Thus $T_{k+1} = P_0$, and $T_t = a_t P_t + (1 - a_t)T_{t+1}$. This definition remains valid when $R = 0$ and does not require division by R.

The weight of a higher-priority exception contains no factor involving the new rule and is unchanged. Every old exception in the suffix, and the default, acquires the factor $1 - b$. The new exception receives weight $\delta = bR$. Therefore

$$P^+_s = H_s + R[(1 - b)T_t + bQ_s].$$

With $K_s = H_s + (1-b)RT_t$, the common mass is $1-\delta$ and

$$P_s = K_s + \delta T_t, \qquad P_s^+ = K_s + \delta Q_s.$$

Subtracting gives $P_s^+ - P_s = \delta(Q_s - T_t)$. Taking half the ℓ_1 norm proves $\mathrm{TV}(P_s^+, P_s) = \delta\,\mathrm{TV}(Q_s, T_t) \le \delta$. For comparison, appending a softmax logit with normalized weight η multiplies every old weight by $1-\eta$, giving $(1-\eta)P_s + \eta Q_s$. □

Affine dependence on the edited node. Collect the traces that contain s. Each contains s exactly once, and its factors outside the subtrace rooted at s involve only variables in $\mathbf{S}^c$. Summing over all choices within that subtrace gives P_s. The traces containing s therefore contribute $A_s(\mathbf{x}_{\mathbf{S}^c})P_s(\mathbf{x}_{\mathbf{S}})$, while those avoiding s contribute a nonnegative function B_s. Consequently,

$$P(\mathbf{x}) = A_s(\mathbf{x}_{\mathbf{S}^c})P_s(\mathbf{x}_{\mathbf{S}}) + B_s(\mathbf{x}), \qquad A_s, B_s \ge 0. \tag{6}$$

Both functions stay fixed during the local edit. Only A_s is restricted to variables in $\mathbf{S}^c$; B_s can depend on all of $\mathbf{X}$. The argument uses traces and therefore includes circuits in which s has several parents.

Marginalizing Eq. (6) under evidence gives

$$P(\mathbf{e}) = A_s(\mathbf{e}_{\mathbf{S}^c})P_s(\mathbf{e}_{\mathbf{S}}) + B_s(\mathbf{e}), \qquad D_s(\mathbf{e}) = \frac{\partial P(\mathbf{e})}{\partial P_s(\mathbf{e}_{\mathbf{S}})} = A_s(\mathbf{e}_{\mathbf{S}^c}).$$

Here evaluating A_s or B_s on a partial assignment means summing over compatible assignments to the unobserved variables. Thus

$$\Delta_{\mathbf{e}} := P^+(\mathbf{e}) - P(\mathbf{e}) = \delta D_s(\mathbf{e})\big[Q_s(\mathbf{e}_{\mathbf{S}}) - T_t(\mathbf{e}_{\mathbf{S}})\big]. \tag{7}$$

Let $\pi_s = \sum_{\mathbf{x}_{\mathbf{S}^c}} A_s(\mathbf{x}_{\mathbf{S}^c})$. Normalization of P_s shows that π_s is the probability that an unconditioned trace visits s, so $0 \le \pi_s \le 1$. The corresponding unconditional identity is

$$\mathrm{TV}(P^+, P) = \pi_s \delta\,\mathrm{TV}(Q_s, T_t) \le \pi_s \delta. \tag{8}$$

Proof of Corollary 7. Subtracting the two instances of Eq. (6) gives

$$P^+(\mathbf{x}) - P(\mathbf{x}) = \delta A_s(\mathbf{x}_{\mathbf{S}^c})\big[Q_s(\mathbf{x}_{\mathbf{S}}) - T_t(\mathbf{x}_{\mathbf{S}})\big].$$

Summing over $\mathbf{x}_{\mathbf{S}}$ gives zero because Q_s and T_t are normalized on $\mathbf{S}$. The marginal on $\mathbf{S}^c$ is therefore unchanged. If $\mathbf{F} \cup \mathbf{E} \subseteq \mathbf{S}^c$, both $P(\mathbf{e} \wedge \mathbf{f})$ and $P(\mathbf{e})$ are unchanged. Their ratio is unchanged whenever $P(\mathbf{e}) > 0$. Evidence involving $\mathbf{S}$ is outside this guarantee. □

Proof of Theorem 8. Set $K(\mathbf{x}) = A_s(\mathbf{x}_{\mathbf{S}^c})K_s(\mathbf{x}_{\mathbf{S}}) + B_s(\mathbf{x})$. Then $K \ge 0$, $P = K + \delta A_s T_t$, and $P^+ = K + \delta A_s Q_s$. Under evidence $\mathbf{e}$, these three parts have masses $r_{\mathbf{e}}$, $v_{\mathbf{e}}$, and $u_{\mathbf{e}}$ as defined in the main text. In particular, $r_{\mathbf{e}} \ge 0$, $Z = r_{\mathbf{e}} + v_{\mathbf{e}} > 0$, and $Z^+ = r_{\mathbf{e}} + u_{\mathbf{e}} > 0$. For each assignment consistent with the evidence,

$$\min\left\{\frac{P(\mathbf{x})}{Z}, \frac{P^+(\mathbf{x})}{Z^+}\right\} \ge \frac{K(\mathbf{x})}{\max\{Z, Z^+\}}.$$

Summing gives an overlap of at least $r_{\mathbf{e}}/\max\{Z, Z^+\}$ between the conditioned distributions. Since total variation is one minus their overlap,

$$\begin{aligned}\mathrm{TV}\big(P^+(\cdot \mid \mathbf{e}), P(\cdot \mid \mathbf{e})\big) &\le 1 - \frac{r_{\mathbf{e}}}{\max\{Z, Z^+\}} \\ &= \max\left\{1 - \frac{r_{\mathbf{e}}}{Z}, 1 - \frac{r_{\mathbf{e}}}{Z^+}\right\} = \max\left\{\frac{v_{\mathbf{e}}}{Z}, \frac{u_{\mathbf{e}}}{Z^+}\right\}.\end{aligned}$$

An upward pass evaluates the original circuit, and a backward pass computes $D_s(\mathbf{e})$. Suffix values follow from the recurrence for T_t; the new component is evaluated separately. The cost is linear in the original circuit and the evaluated new component, after gate and input evaluation. □

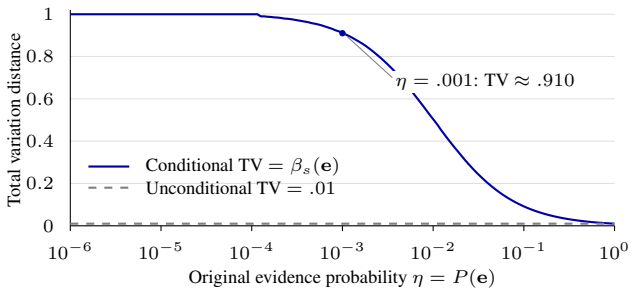

Figure 2: The sharpness construction with $\delta = .01$. The unconditional total variation is .01, while the conditional total variation, equal to the certificate, approaches one as $P(\mathbf{e}) = \eta$ approaches zero.

Sharpness. Let Y, F be binary and let the original distribution have $P(Y = 0, F = 0) = 1 - \eta$ and $P(Y = 1, F = 0) = \eta$, where $0 < \eta < 1$. Insert at the root, above all existing rules, a component concentrated on $(Y = 1, F = 1)$ with applicability $0 < \delta < 1$. Then $P^+ = (1 - \delta)P + \delta Q_s$. For evidence $\mathbf{e} = (Y = 1)$,

$$v_{\mathbf{e}} = \delta\eta, \qquad u_{\mathbf{e}} = \delta, \qquad r_{\mathbf{e}} = (1 - \delta)\eta, \qquad Z^+ = \delta + (1 - \delta)\eta.$$

The original conditional assigns probability zero to $F = 1$, whereas the revised conditional assigns

$$P^+(F = 1 \mid \mathbf{e}) = \frac{\delta}{\delta + (1 - \delta)\eta} = \beta_s(\mathbf{e}).$$

This is their total variation, proving sharpness. Both distributions are smooth, decomposable mixtures of products of point masses. For $\delta = .01$ and $\eta = .001$, the conditional change is approximately .909918, while the unconditional total variation is .01. Figure 2 plots this example. Sharpness is an existence statement: for an individual edit the certificate can be conservative, including when $Q_s = T_t$ and the distribution does not change.

A.3 Conditional Severity and the Audit Budget

Proof of Lemma 9. For either circuit $M \in \{P, Q\}$ and a node s with scope $\mathbf{S}$, define

$$\sigma_{\mathbf{S}}(\mathbf{x}_{\mathbf{S}}) = \sum_{i:X_i \in \mathbf{S}} \sigma_i(x_i), \quad p_s = \sum_{\mathbf{x}_{\mathbf{S}} \models \mathbf{e}_{\mathbf{S}}} M_s(\mathbf{x}_{\mathbf{S}}), \quad m_s = \sum_{\mathbf{x}_{\mathbf{S}} \models \mathbf{e}_{\mathbf{S}}} \sigma_{\mathbf{S}}(\mathbf{x}_{\mathbf{S}}) M_s(\mathbf{x}_{\mathbf{S}}).$$

At a univariate input these are exactly the quantities in the lemma, using ϕ_n and the severity of its variable. At a smooth sum, all children have scope $\mathbf{S}$, so both quantities are linear: $p_s = \sum_c w_{sc}^M p_c$ and $m_s = \sum_c w_{sc}^M m_c$. At a product, the child scopes $\mathbf{S}_c$ partition $\mathbf{S}$ and the severity is their sum. Hence

$$\begin{aligned} m_s &= \sum_{\mathbf{x}_{\mathbf{S}} \models \mathbf{e}_{\mathbf{S}}} \left(\sum_c \sigma_{\mathbf{S}_c}(\mathbf{x}_{\mathbf{S}_c}) \right) \prod_{c'} M_{c'}(\mathbf{x}_{\mathbf{S}_{c'}}) \\ &= \sum_c m_c \prod_{c' \neq c} p_{c'}. \end{aligned}$$

At the root, $m_{\text{root}}/p_{\text{root}} = \mathbb{E}_M[\sigma \mid \mathbf{e}]$. Prefix and suffix products compute all sibling products in linear time without division, including when a child evidence probability is zero. The result also applies to multivariate inputs if their local evidence probabilities and severity moments are tractable. □

Proof of Proposition 10. All expectations in this proof condition on the same evidence $\mathbf{e}$. Linearity gives $G = \sum_i G_i$. For $a > 0$,

$$\mathbb{E}_P[a\sigma + b \mid \mathbf{e}] - \mathbb{E}_Q[a\sigma + b \mid \mathbf{e}] = aG.$$

Thus a positive change of scale multiplies the gap, and the additive constant cancels; signs and rankings are preserved. Let $c = \min_{\mathbf{x}} \sigma(\mathbf{x})$ and suppose first that $L > 0$. Then $g = (\sigma - c)/L$ takes values in $[0, 1]$. Using $\mathbb{E}[g] = \int_0^1 \Pr(g > t)\, dt$ and the event characterization of total variation gives

$$|G| \le L \ \mathrm{TV}\big(P(\cdot \mid \mathbf{e}), Q(\cdot \mid \mathbf{e})\big) \le L\sqrt{\mathrm{KL}\big(Q(\cdot \mid \mathbf{e}) \| P(\cdot \mid \mathbf{e})\big)/2},$$

where the second inequality is Pinsker's inequality (Cover & Thomas, 2006). If $L = 0$, the severity is constant and $G = 0$. A zero gap need not imply equal distributions: on severity levels $0, 1, 2$, a point mass on 1 and equal masses on 0 and 2 have the same expected severity. □

Proof of Proposition 13 and Theorem 12. The circuits P and Q share their graph and input distributions, differing only in sum weights. First suppose that every trace with positive $Q(\cdot \mid \mathbf{e})$ probability also has positive $P(\cdot \mid \mathbf{e})$ probability. Given a trace and the evidence, both models use the same product of conditioned input distributions. The conditional KL for outcomes given the trace is therefore zero. The chain rule gives

$$\begin{aligned}\mathrm{KL}\big(Q(T, \mathbf{X} \mid \mathbf{e}) \| P(T, \mathbf{X} \mid \mathbf{e})\big) &= \mathrm{KL}\big(Q(T \mid \mathbf{e}) \| P(T \mid \mathbf{e})\big) \\ &\quad + \mathbb{E}_{T \sim Q(\cdot \mid \mathbf{e})}\, \mathrm{KL}\big(Q(\mathbf{X} \mid T, \mathbf{e}) \| P(\mathbf{X} \mid T, \mathbf{e})\big) \\ &= \mathrm{KL}\big(Q(T \mid \mathbf{e}) \| P(T \mid \mathbf{e})\big).\end{aligned}$$

By Eq. (5),

$$\begin{aligned}\mathrm{KL}\big(Q(T \mid \mathbf{e}) \| P(T \mid \mathbf{e})\big) &= \mathbb{E}_{T \sim Q(\cdot \mid \mathbf{e})}\left[\sum_{s \in \mathcal{S}(T)} \log \frac{\widehat{w}^Q_{s,T(s)}(\mathbf{e})}{\widehat{w}^P_{s,T(s)}(\mathbf{e})}\right] \\ &= \sum_{s \in \mathcal{S}} \rho^Q_s(\mathbf{e})\, \mathrm{KL}\big(\widehat{w}^Q_s(\mathbf{e}) \| \widehat{w}^P_s(\mathbf{e})\big) = B_{\mathbf{e}}.\end{aligned}$$

Each node appears at most once in a trace. Its contribution is therefore weighted by its probability of being reached, and its child is drawn from the posterior weights conditional on that event. This proves the first identity in Proposition 13.

Expanding the same joint KL in the other order gives

$$\begin{aligned}B_{\mathbf{e}} &= \mathrm{KL}\big(Q(\mathbf{X} \mid \mathbf{e}) \| P(\mathbf{X} \mid \mathbf{e})\big) \\ &\quad + \mathbb{E}_{\mathbf{X} \sim Q(\cdot \mid \mathbf{e})}\, \mathrm{KL}\big(Q(T \mid \mathbf{X}, \mathbf{e}) \| P(T \mid \mathbf{X}, \mathbf{e})\big).\end{aligned}$$

The last term is nonnegative, proving the outcome KL bound. Combining it with Proposition 10 and $\mathrm{TV} \le 1$ gives $|G(\mathbf{e}, \mathbf{u})| \le L \min\{1, \sqrt{B_{\mathbf{e}}/2}\}$.

We define terms with zero Q-flow to be zero, and use the conventions $0 \log(0/p) = 0$ and $q \log(q/0) = +\infty$ for $q > 0$. If a trace has positive Q-posterior but zero P-posterior, set $B_{\mathbf{e}} = +\infty$; the bound remains valid, while the finite-budget decomposition is asserted only when its terms are well defined. With finite descriptive softmax logits and shared inputs, every reference-supported trace is descriptively possible, so the budget is finite whenever both evidence probabilities are positive. Two upward passes provide the node evidence values, and one backward pass through Q provides all Q-flows. The sum over gate KL divergences then takes linear time in the number of edges. □

The worked audit and a shared reference. In Example 11, with empty evidence, $P = (.66, .16, .18)$ and $Q = (.90, .10, 0)$. Thus

$$B_\varnothing = -\log .7 \approx .3567, \qquad \mathrm{KL}(Q \| P) = .9 \log(.9/.66) + .1 \log(.1/.16) \approx .2321.$$

Their difference is approximately $.1245$ nats. The expected severities are $.88$ and $.10$, giving gap $.78$; the mild and severe contributions are $.06$ and $.72$. If the same reference is used for every candidate and the neutral input, its expectation cancels from the score in Section 4:

$$S_\sigma(\mathbf{e}, \mathbf{u}) = \mathbb{E}_{P(\cdot \mid \mathbf{e}, \mathbf{u})}[\sigma] - \mathbb{E}_{P(\cdot \mid \mathbf{e}, \mathbf{u}_0)}[\sigma].$$

Candidate independence requires the entire reference policy to be fixed, including its weights at audited nodes. A candidate-dependent masking policy need not cancel merely because its unaudited gates use a common reference input.

B Other Edits and an Audit Procedure

The certificate uses only a common-part representation. For any local edit of the form $P_s = K_s + \delta V_s$ and $P_s^+ = K_s + \delta Q_s$, with $K_s \geq 0$ and normalized V_s, Q_s on the same scope, the proof of Theorem 8 applies with V_s in place of T_t. Table 3 gives the removed and added components for several edits. Here R is the unchanged prefix mass, T_{i+1} is the normalized suffix below rule i, and $j = i + 1$ in an adjacent swap.

Deleting rule i replaces $H_s + R[a_i P_i + (1-a_i)T_{i+1}]$ by $H_s + RT_{i+1}$, with common part $H_s + R(1-a_i)T_{i+1}$. An applicability change has common part $H_s + R[\min(a_i, a_i')P_i + (1 - \max(a_i, a_i'))T_{i+1}]$. For a swap, each rule keeps its applicability. The old and new contributions of the pair and its suffix are

$$\begin{aligned} \text{old: } & R[a_i P_i + (1-a_i)a_j P_j + (1-a_i)(1-a_j)T_{j+1}], \\ \text{new: } & R[a_j P_j + (1-a_j)a_i P_i + (1-a_i)(1-a_j)T_{j+1}]. \end{aligned}$$

The new contribution minus the old is $Ra_i a_j(P_j - P_i)$. Their common part, including the prefix, is

$$H_s + R[a_i(1-a_j)P_i + (1-a_i)a_j P_j + (1-a_i)(1-a_j)T_{j+1}].$$

It is nonnegative and has mass $1 - Ra_i a_j$, establishing the representation in Table 3.

Exact revised queries. For compatible evidence and query assignments, apply Eq. (7) to $\mathbf{e}$ and $\mathbf{e} \wedge \mathbf{f}$. This gives

$$P^+(\mathbf{f} \mid \mathbf{e}) = \frac{P(\mathbf{e} \wedge \mathbf{f}) + \Delta_{\mathbf{e}\wedge\mathbf{f}}}{P(\mathbf{e}) + \Delta_{\mathbf{e}}}. \tag{9}$$

Subtracting $P(\mathbf{f} \mid \mathbf{e})$ yields

$$P^+(\mathbf{f} \mid \mathbf{e}) - P(\mathbf{f} \mid \mathbf{e}) = \frac{\Delta_{\mathbf{e}\wedge\mathbf{f}} - P(\mathbf{f} \mid \mathbf{e})\Delta_{\mathbf{e}}}{P(\mathbf{e}) + \Delta_{\mathbf{e}}}.$$

Both conditionals are zero if the assignments conflict. For the other edits in Table 3, replace T_t by V_s in the evidence-change formula.

Decision guarantees. Total variation bounds every event-probability change. A threshold decision at τ therefore cannot change when $|P(\mathbf{f} \mid \mathbf{e}) - \tau| > \beta_s(\mathbf{e})$. For two candidate inputs, write $q_a = P(\mathbf{f} \mid \mathbf{e}, \mathbf{u}_a)$ and $q_b = P(\mathbf{f} \mid \mathbf{e}, \mathbf{u}_b)$, with certificates β_a and β_b computed at those inputs. Then $q_a^+ - q_b^+ \geq q_a - q_b - \beta_a - \beta_b$, so $q_a - q_b > \beta_a + \beta_b$ preserves their order. The neutral term in $S_{\mathbf{f}}$ cancels from this pairwise difference, even if that shared term changes under the revision.

For a severity function of range L, the same bounded-function argument used for Proposition 10 gives

$$\left|\mathbb{E}_{P^+(\cdot|\mathbf{e})}[\sigma] - \mathbb{E}_{P(\cdot|\mathbf{e})}[\sigma]\right| \leq L\beta_s(\mathbf{e}).$$

Thus a pairwise expected-severity margin greater than $L(\beta_a + \beta_b)$ is preserved. A shared normative reference cancels from the pairwise comparison. If the two candidates instead have distinct references that also change, with certificates β_a^Q, β_b^Q, a sufficient margin is $L(\beta_a + \beta_b + \beta_a^Q + \beta_b^Q)$. For an individual audit gap, changing both descriptive and reference distributions gives $|G^+ - G| \leq L(\beta^P + \beta^Q)$.

Table 3: Local edits with a common-part representation.

Edit	Moved mass δ	Removed V_s	Added Q_s
Insert before t	Rb	T_t	new component
Delete rule i	Ra_i	P_i	T_{i+1}
Increase a_i to a_i'	$R(a_i' - a_i)$	T_{i+1}	P_i
Decrease a_i to a_i'	$R(a_i - a_i')$	P_i	T_{i+1}
Swap adjacent $i \prec j$	$Ra_i a_j$	P_i	P_j

Sequences of edits. Let $P^{(0)}, \ldots, P^{(m)}$ be successive models and assume that $\mathbf{e}$ has positive probability under each. The triangle inequality gives

$$\mathrm{TV}\big(P^{(m)}(\cdot \mid \mathbf{e}), P^{(0)}(\cdot \mid \mathbf{e})\big) \le \min\left\{1, \sum_{j=1}^{m} \beta^{(j)}(\mathbf{e})\right\}.$$

Each certificate compares consecutive models at the same exogenous input. It must be recomputed against the current model.

Audit procedure. For a proposed local edit and a fixed $(\mathbf{e}, \mathbf{u})$:

1. Evaluate the original circuit under $\mathbf{e}$ and compute $Z = P(\mathbf{e})$. Require $Z > 0$.
2. Run a backward pass to obtain $D_s(\mathbf{e})$.
3. Evaluate the removed and added components on $\mathbf{e}_\mathbf{S}$ and compute δ.
4. Compute $v_\mathbf{e}, u_\mathbf{e}, r_\mathbf{e}$ and $Z^+ = r_\mathbf{e} + u_\mathbf{e}$. Require $Z^+ > 0$.
5. Return $\beta_s(\mathbf{e})$ and apply the desired margin criterion. If necessary, repeat the calculation for $\mathbf{e} \wedge \mathbf{f}$ to obtain the exact query.

One backward pass gives derivatives at every node, so alternative edits of the same model can share it for the same evidence and input. Adding a child at an audited DePC node also requires its compliance label and a reference policy on the revised child set. Changes to that reference are certified separately; its certificate is not automatically the descriptive certificate.

C Conditional Obligations

We give an explicit circuit construction for two reference policies: one for ideal behavior and one for the obligations applicable after a reported violation. Let binary variables A, B describe the policy's targets. The primary target is $A = 1$; the conditional target is $B = 1$ when $A = 1$, and $B = 0$ when $A = 0$. The observed report $u_a = 0$ is exogenous information indicating that the primary requirement was violated. It is distinct from conditioning the reference target variable A on zero.

Let q_A, q_1, q_0 be normalized Bernoulli distributions satisfying $q_A(1) \ge 1 - \epsilon$, $q_1(1) \ge 1 - \epsilon$, and $q_0(0) \ge 1 - \epsilon$, for $0 \le \epsilon < 1/4$. The reference triggered by the report is

$$Q_{\mathrm{act}}(a, b \mid u_a = 0) = q_A(a) q_0(b). \tag{10}$$

It retains the primary target and prescribes the remedial target: $Q_{\mathrm{act}}(A = 1 \mid u_a = 0) \ge 1 - \epsilon$ and $Q_{\mathrm{act}}(B = 0 \mid u_a = 0) \ge 1 - \epsilon$. It does not assert that the primary target was achieved.

The ideal-case reference instead uses a structural case split:

$$Q_{\mathrm{ideal}}(a, b) = q_A(1) \mathbb{1}\{a = 1\} q_1(b) + q_A(0) \mathbb{1}\{a = 0\} q_0(b). \tag{11}$$

Each child of the root sum is a product on disjoint scopes $\{A\}$ and $\{B\}$, and both products have scope $\{A, B\}$. The sum weights are $q_A(1), q_A(0)$, so the circuit is normalized, smooth, and decomposable. It satisfies

$$Q_{\mathrm{ideal}}(A = 1) \ge 1 - \epsilon, \qquad Q_{\mathrm{ideal}}(B = 1) \ge (1 - \epsilon)^2 > 1/2.$$

These two policies can share a circuit: at the $A = 1$ branch, include a sum over q_1 and q_0. The ideal policy selects q_1 there, whereas the report-triggered policy selects q_0; the $A = 0$ branch uses q_0 in both cases. The graph and input distributions remain fixed and only the reference weights change.

Under a probability-threshold reading with threshold above $1/2$, neither reference assigns both $B = 1$ and $B = 0$ the status of a required target. The construction distinguishes ideal and report-triggered targets by an explicit policy choice. It uses a structural split for A because A is modeled inside the circuit; using A as an externally fixed gate input would define a different conditional model. This is a probabilistic representation of the supplied policies, rather than a derivation of the policies from descriptive data.

D LEARNING DETAILS

Likelihood gradients and posterior flows. For an observation $(\mathbf{x}, \mathbf{u})$, fix $\mathbf{u}$ and let $v_s = P_s(\mathbf{x}_{\mathrm{sc}(s)} \mid \mathbf{u})$ and $d_s = \partial v_{\mathrm{root}} / \partial v_s$. Assume $v_{\mathrm{root}} > 0$. The posterior probability of traversing sum edge (s, c) is

$$\gamma_{sc} = \frac{d_s w_{sc} v_c}{v_{\mathrm{root}}}, \qquad \rho_s = \sum_c \gamma_{sc} = \frac{d_s v_s}{v_{\mathrm{root}}}. \tag{12}$$

The ratio $w_{sc} v_c / v_s$, when defined, conditions on reaching s. Multiplying it by ρ_s accounts for the probability of reaching the node. For a softmax logit, the exact likelihood derivative is

$$\frac{\partial \log P_\theta(\mathbf{x} \mid \mathbf{u})}{\partial z_{sc}} = \gamma_{sc} - \rho_s w_{sc}. \tag{13}$$

This follows by combining $\partial \log P / \partial w_{sc} = d_s v_c / v_{\mathrm{root}}$ with the softmax derivative. Gradients through shared gate encoders sum all such contributions. The same formulas apply to incomplete observations after replacing the complete assignment by evidence.

For an EM step, compute responsibilities under the current parameters. Holding them fixed, the gate part of the auxiliary objective is

$$\sum_n \sum_{s \in \mathcal{S}} \sum_{c \in \mathrm{ch}(s)} \gamma_{sc}^{(n)} \log w_{sc}(\mathbf{u}^{(n)}; \theta).$$

Input parameters use the corresponding posterior input flows. Shared encoders couple the gate terms, so a generalized M-step optimizes them jointly; exact responsibilities do not imply a closed-form optimum for a neural gate. The experiments described in Section E.2 use gradient optimization.

Optional entropy regularization. An entropy penalty added to a maximized likelihood objective has the form $-\lambda \sum_{n,s} H(w_s(\mathbf{u}^{(n)}))$, with $\lambda \geq 0$ and $H(w) = -\sum_i w_i \log w_i$. It favors concentrated gates. The following property concerns routing concentration and does not identify branch semantics.

Proposition 14 (Entropy bounds routing)**.** *Let $P_s = \sum_i w_i P_i$, let $i^* \in \arg\max_i w_i$, and suppose $H(w) \leq \epsilon$ nats. Then $\mathrm{TV}(P_s, P_{i^*}) \leq 1 - w_{i^*} \leq 1 - e^{-\epsilon}$.*

Proof. By the triangle inequality,

$$\mathrm{TV}(P_s, P_{i^*}) \leq \sum_{i \neq i^*} w_i \, \mathrm{TV}(P_i, P_{i^*}) \leq 1 - w_{i^*}.$$

Since $w_i \leq w_{i^*}$, $H(w) = \sum_{i: w_i > 0} w_i \log(1/w_i) \geq \log(1/w_{i^*})$. Thus $w_{i^*} \geq e^{-H(w)} \geq e^{-\epsilon}$. □

Learning a new exception with protected contexts. The local-learning extension in Section 5 keeps all pre-existing components and gate parameters fixed and learns only a new normalized component $Q_s(\cdot \mid \mathbf{u}; \varphi)$ and its applicability $b(\mathbf{u}; \psi) \in [0, 1]$. Let $\mathcal{D}_{\mathrm{new}}$ be the new observations and $\mathcal{C}$ a finite set of protected context-input pairs $(\mathbf{e}, \mathbf{u})$, with tolerances $\epsilon_{\mathbf{e},\mathbf{u}} \geq 0$. One can fit the new exception by

$$\begin{aligned} \min_{\varphi, \psi} \quad & - \sum_{(\mathbf{x},\mathbf{u}) \in \mathcal{D}_{\mathrm{new}}} \log P^+_{\varphi,\psi}(\mathbf{x} \mid \mathbf{u}) \\ \text{subject to} \quad & \beta_s(\mathbf{e}; \mathbf{u}) \leq \epsilon_{\mathbf{e},\mathbf{u}}, \qquad (\mathbf{e}, \mathbf{u}) \in \mathcal{C}, \end{aligned} \tag{14}$$

requiring positive original and revised evidence probabilities on each protected pair. The notation $\beta_s(\mathbf{e}; \mathbf{u})$ makes explicit the input suppressed in Section 3. At every feasible solution, Theorem 6 preserves all higher-priority rules, and Theorem 8 gives

$$\left| P^+_{\varphi,\psi}(\mathbf{f} \mid \mathbf{e}, \mathbf{u}) - P(\mathbf{f} \mid \mathbf{e}, \mathbf{u}) \right| \leq \epsilon_{\mathbf{e},\mathbf{u}}$$

for every query $\mathbf{f}$ at every protected pair. These are guarantees at the constrained contexts. A soft penalty on the certificates can guide optimization, but the specified tolerances are guaranteed only when the constraints are satisfied.

E EXPERIMENTAL DETAILS

E.1 DATASET CONSTRUCTION

Physical dataset. The main physical dataset contains 32 categorical world variables, of which 26 may appear in ranking contexts. Six intermediate or outcome variables, *Stability*, *Motion*, *Collision*, *Damage*, *Spill*, and *FinalPosition*, are excluded from ranking contexts to prevent outcome leakage. The update space contains 109 update identities, including a neutral update. We use 100,000 observations for training and 10,000 for validation. The ranking test set contains 1,080 instances with 8 candidate updates per instance. Context sizes are 16, 20, and 24, with 360 instances at each size. The three hypotheses, *Motion=fall*, *Damage=broken*, and *Spill=full*, are equally represented. Gold rankings are generated by the frozen ground-truth physical model and are used only for evaluation.

Clinical dataset. The main clinical dataset contains 32 categorical world variables and 8 structured action variables defining 31 action configurations, including a neutral action. We use 100,000 observations for training and 10,000 for validation. The ranking test set contains 1,080 instances with 8 candidate actions per instance. Context sizes are 16, 20, and 24, with 360 instances at each size. Gold rankings are generated by the frozen ground-truth clinical model and are used only for evaluation.

Generalization datasets. We additionally construct independent physical and clinical generalization datasets using separately specified stochastic simulators. The physical generalization dataset is generated by a hand-specified stochastic Bayesian-network model rather than a probabilistic circuit. It contains 14 upstream variables, 4 outcome variables, 6 update variables, and 17 update identities including neutral. We use 50,000 observations for training and 5,000 for validation. The ranking test set contains 1,080 instances with context sizes 10, 12, and 14. The clinical generalization dataset is generated by an independent stochastic rule simulator rather than the ground-truth DePC. It retains the observable clinical variables and the 31-action catalog from the main clinical setting. We use 100,000 observations for training and 10,000 for validation. The ranking test set contains 1,080 instances with context sizes 16, 20, and 24. Gold rankings are generated directly by the independent simulator.

E.2 TRAINING AND IMPLEMENTATION DETAILS

Circuit training. All circuit models are trained on the same observational training split within each domain. Ranking labels are used only for evaluation and never for training or model selection. Checkpoints are selected using validation negative log-likelihood. We use AdamW with a batch size of 512, weight decay of 10^{-5}, and gradient clipping at 5.0. Models are trained for at most 50 epochs with early-stopping patience of 8. No learning-rate scheduler or mixed-precision training is used. DPC and DePC use 8 channels, 4 replicas, and 8 semantic branches. The PC learning rate is 10^{-3}. For DPC, the gate learning rate is also 10^{-3}; for DePC, it is 3×10^{-3}. We set the random seed as 0.

Model capacity. CSPN is configured to closely match the trainable parameter count of the corresponding DPC or DePC model. Vanilla PC is included as a structural baseline and is not capacity matched. Approximate parameter counts are reported in the main results table.

Language-model baselines. The language-model baselines receive the same partial context and 8 candidate descriptions as the corresponding ranking instance, but no training observations or gold ranking scores. GPT-5.6 Sol is evaluated with medium reasoning effort and a maximum output length of 4096 tokens. Qwen3.8-27B and Qwen3-32B are evaluated locally. Qwen3.8-27B uses deterministic inference with a maximum of 256 generated tokens and no-think mode.

E.3 EVALUATION DETAILS

Clinical scoring. For the clinical experiments, all learned probabilistic circuit models use the same fixed severity utility over *AdverseEvent*, *PatientSafety*, and *ProtocolStatus*. For a joint outcome assignment $\mathbf{y}$ to these three outcome variables,

$$\sigma(\mathbf{y}) = 0.5\,\sigma_{\mathrm{adv}}(\mathbf{y}) + 0.25\,\sigma_{\mathrm{safety}}(\mathbf{y}) + 4.0\,\sigma_{\mathrm{protocol}}(\mathbf{y}), \tag{15}$$

where adverse-event levels receive scores $0, 1, 3$, patient-safety levels receive scores $0, 1, 2$, and the protocol states *compliant*, *authorized*, *minor violation*, and *major violation* receive scores $0, 0, 2, 3$, respectively. Expectations are computed exactly over the 36 joint outcome assignments. The same utility is used for DePC, CSPN, and Vanilla PC. The range of σ is $[0, 14]$, so $L = 14$ in the bounds of Sections 3 and 4.

Metrics. Strict Kendall's τ compares the complete predicted ranking with the strict ground-truth ranking and is averaged over queries. Tie-aware Kendall's τ excludes candidate pairs whose ground-truth score difference is below an ambiguity threshold before computing ranking agreement. Pairwise accuracy is the pooled fraction of comparable candidate pairs ordered correctly. Top-1 accuracy measures whether the highest-ranked candidate matches the ground truth, while exact-ranking accuracy requires the complete 8-candidate ordering to match.

Difficulty groups. For the physical benchmarks, candidate pairs with a gold-score gap below 0.003 are treated as ambiguous. Hard, Medium, and Easy pairs are defined by

$$[0.003, 0.01), \qquad [0.01, 0.03), \qquad [0.03, \infty),$$

respectively.

For the clinical benchmarks, candidate pairs with a gold-score gap below 0.01 are treated as ambiguous. Hard, Medium, and Easy pairs are defined by

$$[0.01, 0.05), \qquad [0.05, 0.20), \qquad [0.20, \infty),$$

respectively.

Bootstrap evaluation. For the clinical comparisons between DePC and CSPN, we perform 10,000 paired bootstrap resamples over ranking queries. The same query indices are sampled for both models in each replicate. We report percentile 95% confidence intervals using the 2.5th and 97.5th percentiles of the bootstrap distribution. These intervals condition on the fitted models and quantify variation over the sampled queries; they do not include variation over training runs.

F Additional Results

F.1 Results by Difficulty Group

Table 4: Pairwise accuracy (%) across difficulty levels on the physical benchmarks.

Dataset	Model	Hard	Medium	Easy
Main	DPC (ours)	**57.34**	**67.48**	**89.31**
	CSPN	51.50	58.03	84.20
	Vanilla PC	49.92	55.10	78.49
	GPT-5.6 Sol	48.50	52.70	72.00
	Qwen3.8-27B	47.20	51.65	72.90
	Qwen3-32B	47.79	51.84	69.89
Generalization	DPC (ours)	60.06	**81.05**	**94.57**
	CSPN	**60.75**	73.78	93.27
	Vanilla PC	55.07	79.19	91.80

As shown in Table 4, DPC exceeds CSPN at all three difficulty levels on the main benchmark. On the generalization benchmark it leads on Medium and Easy pairs, while CSPN is marginally higher on Hard pairs (60.75% versus 60.06%).

Table 5 shows that DePC and CSPN are close on Hard pairs in the main benchmark, while DePC achieves higher accuracy on Medium and Easy pairs. On the generalization benchmark, DePC is higher across all three difficulty levels.

Table 5: Pairwise accuracy (%) across difficulty levels on the clinical benchmarks.

Dataset	Model	Hard	Medium	Easy
Main	DEPC (ours)	57.74	**70.84**	**91.55**
	CSPN	**58.18**	70.20	90.86
	Vanilla PC	50.53	55.50	84.34
	GPT-5.6 Sol	52.86	55.63	83.05
	Qwen3.8-27B	51.01	51.63	82.34
	Qwen3-32B	52.51	51.34	81.59
Generalization	DEPC (ours)	**79.67**	**69.20**	**95.39**
	CSPN	77.34	67.05	95.05
	Vanilla PC	76.29	59.16	87.51

Table 6: Paired bootstrap comparison between DePC and CSPN using 10,000 query-level resamples. Differences are DePC minus CSPN. Pairwise-accuracy differences and intervals use the $[0, 1]$ scale; for example, 0.0059 is 0.59 percentage points.

Dataset	Metric	Difference	95% CI
Main	Strict τ	+0.0112	[0.0017, 0.0206]
Main	Pairwise accuracy	+0.0059	[0.0013, 0.0106]
Generalization	Strict τ	+0.0108	[0.0050, 0.0166]
Generalization	Pairwise accuracy	+0.0053	[0.0029, 0.0078]

F.2 Bootstrap Confidence Intervals

As shown in Table 6, the estimated DePC–CSPN differences are positive for both strict Kendall's τ and pairwise accuracy on the main and generalization benchmarks. The reported generalization difference of 0.0108 is based on unrounded scores; subtracting the rounded strict-τ values in the main table gives 0.0107.

G Example ranking instances.

The following examples illustrate the inputs presented in the ranking evaluation.

Physical benchmark (DPC).
Context: Material: glass; size: large; lid: open; surface friction: medium; surface slope: steep; adhesion: none; surface hardness: soft; surface: damp; force strength: strong; force direction: toward edge; force duration: sustained; vibration: none; tilt change: large; obstacle distance: near; obstacle mass: medium; obstacle height: medium.
Hypothesis: The object will fall.
Candidate updates:
A. The surface becomes substantially steeper.
B. A new inspection reveals that `SupportQuality = unstable`.
C. The obstacle becomes much harder.
D. A strong push is applied to the object.
E. Gravity becomes substantially stronger.
F. A new inspection reveals that `ClearanceToEdge = near`.
G. An obstacle moves close to the object's path.
H. The surface friction increases substantially.
Task: Rank all eight updates from the update that most strengthens the hypothesis to the update that most weakens it, considering the full context.

Clinical benchmark (DePC).
Context: Pediatric patient; sedative medication; renal function: severely impaired; order status: absent; emergency status: urgent; override authorization: denied; monitoring unavailable.
Candidate actions:
A. Administer before dose verification and without an independent double-check.
B. Use a procedural override.
C. Administer using the standard order on record.
D. Administer before identity verification.
E. Administer before dose verification, without a double-check, and without monitoring.
F. Administer before checking the order.
G. Administer before checking potential drug interactions.
H. Administer before verifying the route.
Task: Rank the eight candidate actions from the greatest to the smallest protocol deviation.