LLM Explainability with Counterfactual Chains and Causal Graphs

Anonymous ACL submission

Abstract

Causal graphs provide a high-level language for making mechanisms transparent. Recent work uses Large Language Models (LLMs) to recover causal graphs of external-world processes. Instead, in this paper, we use causal graphs to model LLM inference itself, providing stakeholders with a transparent view of how the model perceives and organizes high-level concepts to produce a prediction. We propose a four-phase method for constructing such graphs. Given a target LLM and a set of textual examples, our method discovers class-discriminative, human-interpretable concepts and maps each input to LLM-perceived concept states. We then introduce an MCMC-inspired counterfactual augmentation procedure that expands the sparse observational data through chains of counterfactuals. This enables stable causal discovery with σ-CG, yielding informative, interpretable graphs. We apply our method to three LLMs across disease diagnosis, sentiment analysis, and LLM-as-a-judge classification tasks. We evaluate the learned graphs for predictive fidelity and structural stability, and the MCMC-inspired augmentation for convergence and downstream utility. Our results show that the discovered causal graphs capture meaningful dependencies consistent with LLMs' reasoning. Together, this paper provides a foundation for concept-level explainability of LLMs.

1 Introduction

Large Language Models (LLMs) exhibit broad capabilities (Bubeck et al., 2023; Brown et al., 2020), yet their inference remains opaque: decision factors are unobservable, and generated explanations often lack faithfulness (Feder et al., 2022; Turpin et al., 2023; Gat et al., 2024; Zhang et al., 2024; Lyu et al., 2023; Lanham et al., 2023). This opacity hinders adoption in high-stakes domains, risking confident hallucinations and biases in medicine (Omiye et al., 2024), and fictitious precedents that undermine legal accountability (Dahl et al., 2024).

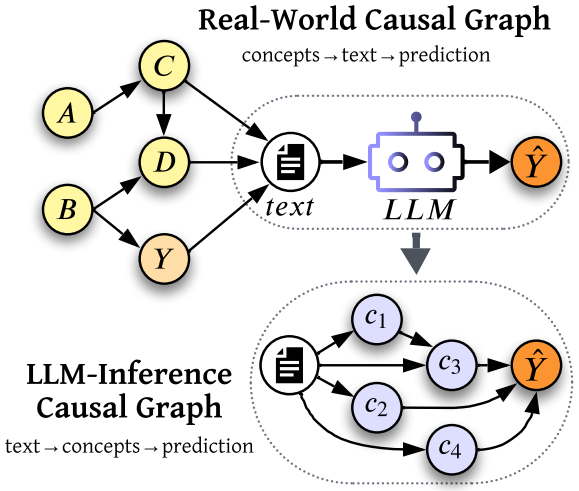


Figure 1: Causal graphs provide a language for modeling mechanisms. Prior work studies real-world structures, using text to infer relations among real-world concepts or estimating how interventions on them (e.g., changing an author's gender) affect model predictions. In contrast, we study the LLM-inference causal graph: a causal graph over LLM-perceived concepts that explains how the model maps text to prediction.

LLM opacity has motivated a broad literature on explainability and interpretability (Calderon and Reichart, 2025). Attribution and attention-based methods are primarily correlational, whereas faithful explanations require causal evidence (Jain and Wallace, 2019; Zhao et al., 2023; Zečević et al., 2023). Counterfactual methods provide stronger causal evidence, and recent approaches use causal graphs over textual variables (Wu et al., 2021; Gat et al., 2024; Toker et al., 2026). However, they typically remain local and input-centered: they can quantify the causal effect of a given factor on the prediction but do not recover the model's high-level inference. Mechanistic interpretability targets internal components such as neurons and circuits. While this provides a valuable low-level science of neural computation, it is often misaligned with the explanations stakeholders need (Calderon and

Reichart, 2025; Somvanshi et al., 2026). What remains missing is an applicable explainability method that provides a global, causal account of how a model organizes human-interpretable concepts and reduces them to a prediction.

In this paper, we take a step toward this "holy grail" by proposing a fully automated method that combines concept discovery with causal discovery to construct concept-level causal graphs of LLM inference for classification tasks (Figure 1). A causal graph is a directed graph whose edges encode direct cause-and-effect relationships among variables (Pearl, 2000). In contrast to prior work that uses causal graphs to model relations among real-world variables, we use causal graphs as an interpretability object for the LLM's inference process. Specifically, we construct a graph over the input text, LLM-perceived concept states, and the prediction, which serves as the terminal node. Accordingly, the graph makes explicit how the model organizes concept-level information and how this structure leads to the final prediction.

Our proposed method is model-driven: the target LLM itself identifies latent concepts and generates the data required for causal discovery. Given a set of textual examples and a classification task, our framework implements a four-phase pipeline (Figure 2): (1) Use the LLM to Generate class predictions for the textual examples;(2) extracting concepts that differentiate between these predicted classes and representing each textual example with a concept vector; (3) densely populating the concept space via MCMC-inspired counterfactual data expansion; and (4) constructing a causal graph over the concepts and the model prediction.

A central challenge in causal discovery is data coverage: reliable structure learning requires a representative sample that spans diverse combinations of concept values. In many domains, obtaining such coverage requires costly, impractical, or unethical interventions. Our setting makes this bottleneck more tractable: because the mechanism under study is an LLM, we can prompt the model to generate counterfactual texts in which a targeted concept is altered while the remaining context is kept as stable as possible. Our MCMC-inspired algorithm iteratively proposes counterfactual chains and rejects unfaithful candidates, producing a denser, higher-quality concept space for robust causal discovery.

We evaluate the framework with three LLMs on three classification tasks spanning natural and synthetic datasets: sentiment analysis, disease diagnosis, and LLM-as-a-judge classification. As no ground-truth graphs or baselines exist for this latent process, we evaluate predictive fidelity and expansion utility: we test whether discovered parents best predict each node, and if expansion improves graph accuracy and stability over the original data.

Our experiments reveal a dichotomy in LLM reasoning: on structured synthetic tasks, models converge on similar explanatory concepts, whereas on naturalistic data, each model develops distinct latent heuristics. We further show that the MCMC-inspired expansion reaches distributional and topological convergence, and that the discovered causal parents are stronger predictors of each node than alternative concept sets.

In summary, this work makes three primary contributions. First, we introduce a causal paradigm for global, concept-based explainability that maps LLM reasoning to causal graphs. Second, we propose an MCMC-inspired data augmentation framework that generates counterfactuals to better cover the concept space, enabling robust causal discovery. Third, we develop an evaluation protocol for assessing both the structural stability and predictive fidelity of the resulting causal graphs.

2 Related Work

Causal Graphs and Interpretability. LLM interpretability methods include feature attribution (Qiu et al., 2021; Lan et al., 2025), attention analysis (Yang et al., 2024; Yeh et al., 2024), probing (Kissane et al., 2025; Zheng et al., 2025; Sharma et al., 2025), concept-based explanations (Kim et al., 2018; Zhang et al., 2025), and chain-of-thought rationales (Sanwal, 2025). These methods are fundamentally associative. Faithful explanations require causal evidence (Agarwal et al., 2024), motivating causal graphs as an explanatory formalism. Causal graphs appear in LLM interpretability in three distinct senses.

First, in mechanistic interpretability, causal graphs are defined over model-internal components, such as attention heads, neurons, residual-stream directions, or higher-level representations (Wang et al., 2023). These graphs abstract away from the complete computation and retain only the components hypothesized to causally mediate a targeted behavior (Mueller et al., 2024).

Second, causal graphs are used to enable estimation of the causal effects of high-level concepts on model predictions, typically via causal infer-

ence methods such as counterfactuals (Toker et al., 2026), matching (Gat et al., 2024), or adjustment (Feder et al., 2021). The graphs are usually assumed in advance (e.g., provided by a domain expert) and describe the data-generating process from concepts to text to model prediction, i.e., concepts $\rightarrow$ text $\rightarrow$ prediction (Paul et al., 2024).

Finally, we introduce a *third* utility, causal graphs to describe the model's high-level reasoning: text $\rightarrow$ concepts $\rightarrow$ prediction. In this setting, the graph is itself the explanation, and both the relevant concepts and their causal relations must be inferred rather than assumed. Our work operates in this setting, and although this utility is promising, it has received no attention in the literature.

Causal Discovery and LLMs. Causal discovery aims to recover causal structure from data (Pearl, 2000). but is underdetermined from observational data alone: many graphs can induce the same distribution, requiring assumptions such as acyclicity, temporal ordering, or restricted functional forms (Glymour et al., 2019). Classical methods include constraint-based, score-based, and hybrid approaches (Spirtes et al., 2000; Chickering, 2002; Zanga et al., 2022), with deep learning extensions for non-linear settings (Zheng et al., 2018; Yu et al., 2019). We employ the σ-CG algorithm (Forré and Mooij, 2018), which handles discrete variables and potentially cyclic structures, fitting our setting where we cannot impose prior structural or parametric assumptions on how the LLM relates concepts during inference.

Recent work incorporates LLMs into the causal discovery process, but with a different goal from ours. Since causal graphs are fundamental to scientific modeling, these works use LLMs to help discover causal graphs of real-world mechanisms (Li et al., 2024). Typically, the term *causal discovery with LLMs* refers to using LLMs to predict whether a causal edge exists between two real-world variables (Ma, 2025). While promising, various studies have shown that LLMs often memorize high-frequency relations rather than exhibit genuine causal generalization (Feng et al., 2025).

The closest work to ours is COAT (Liu et al., 2024), which also uses an LLM to propose concepts and then applies causal discovery. However, COAT aims to recover real-world causal structures and, therefore, focuses on local Markov blankets for causal effect estimation (Aliferis et al., 2010). In contrast, our goal is to characterize the model's internal reasoning, which requires two key algorithmic departures. First, we recover the full graph from text to prediction, rather than only a local Markov blanket, to capture the complete reasoning process. Second, because observational data sparsely cover the model's internal decision space, we introduce an MCMC-inspired counterfactual augmentation procedure that actively expands the concept space for more robust graph discovery.

3 Method

We propose a fully automated method that recovers a concept-level causal graph capturing how a target LLM derives its predictions on a classification task. The method is fundamentally model-driven: the same target LLM extracts latent concepts, represents examples by concept vectors, and generates counterfactuals; thus, the resulting graph reflects the model's internal reasoning rather than an external-world process.

Our method, illustrated in Figure 2, employs a four-phase pipeline: **Label Prediction** substitutes ground-truth labels with the LLM's own predictions. **Differentiative Concept Extraction** iteratively identifies a set of concepts that differentiate between predicted classes, representing each example as a concept vector. **MCMC-Inspired Data Expansion** populates the concept space with counterfactuals to ensure coverage. Finally, **Causal Discovery** constructs the final graph over these concepts and the model prediction.

3.1 Problem Statement and Formulation

Given a target LLM f acting as a classifier and a dataset $\mathcal{D} = \{(x^j, y^j)\}_{j=1}^{N}$ of textual examples, split into $\mathcal{D}_{\text{train}}$ and $\mathcal{D}_{\text{test}}$, where x represents the input text and $y \in \mathcal{Y}$ represents the ground-truth label (with $\mathcal{Y}$ being the set of task classes), our goal is to recover a concept-level causal graph that explains how f derives its prediction $\hat{y} = f(x)$.

We adopt the Causal Graph framework of Pearl (2000), relaxed to admit cyclic dependencies (Bongers et al., 2021), acknowledging potential reciprocal influences among the concepts an LLM uses internally. Let $\mathcal{C} = \{c_1, \ldots, c_n\}$ denote a set of human-interpretable concepts. We define a concept-extraction function ϕ that maps an input text x to a *concept vector* of length n:

$$\phi(x) = [l_{c_1}, l_{c_2}, \ldots, l_{c_n}]$$

where each l_{c_i} represents the LLM-perceived state of concept c_i in the context of x. Specifically,

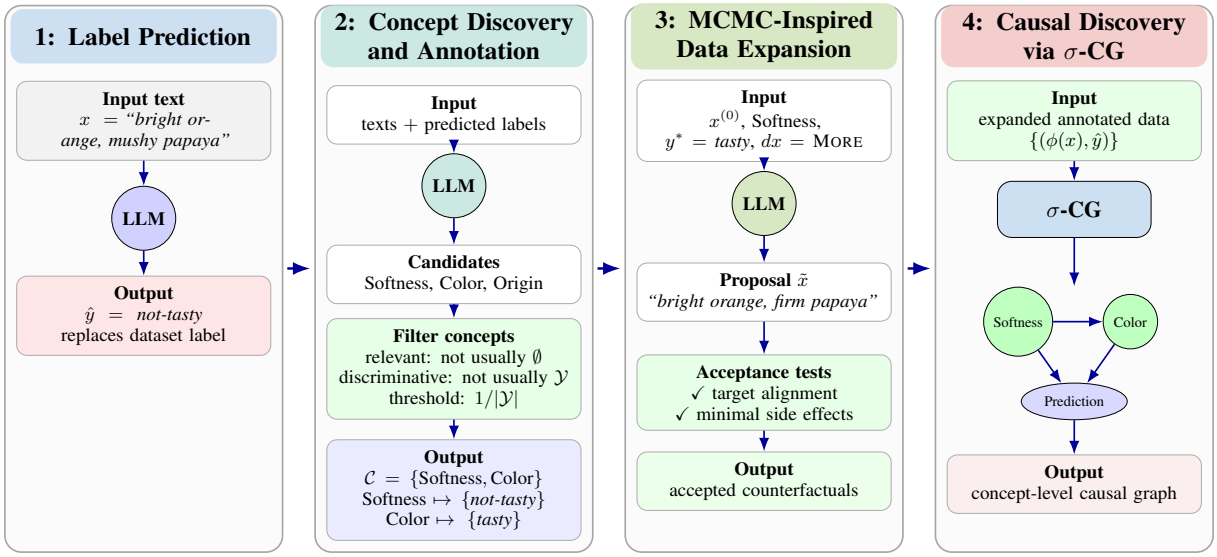


Figure 2: Overview of our four-phase pipeline for constructing causal graphs. The "papaya" running example is drawn from the full toy walkthrough in Appendix B.

l_{c_i} indicates whether the concept is absent and, if present, which task classes the model perceives it to support. We represent each l_{c_i} as a categorical variable taking values in $\mathcal{V} = \{0, 1, \ldots, 2^{|\mathcal{Y}|} - 1\}$, where each value corresponds bijectively to a subset $S \subseteq \mathcal{Y}$. The subset S denotes the classes with which concept c_i is perceived to be aligned. The value corresponding to $S = \emptyset$ indicates that the concept is absent from x or irrelevant, while the value corresponding to $S = \mathcal{Y}$ indicates that the concept is present and aligns with all possible classes without differentiation.

For example, in a disease diagnosis task with $\mathcal{Y} = \{\text{Migraine}, \text{Sinusitis}, \text{Influenza}\}$, suppose c_1 denotes the concept *Headache*. If $\phi(x)[c_1]$ takes the value corresponding to $S = \{\text{Migraine}, \text{Sinusitis}\}$, then the model perceives this concept as present in x and as evidence for both Migraine and Sinusitis. If it takes the value corresponding to $S = \emptyset$, the concept is perceived as absent; if it takes the value corresponding to $S = \mathcal{Y}$, the concept is perceived as present but non-discriminative among the candidate labels.

Finally, we represent the LLM's inference process as a directed graph over the input text, concept variables, and final prediction:

$$\mathcal{G} = (V, E), \qquad V = \{X\} \cup \mathcal{C} \cup \{\hat{Y}\}.$$

The graph captures a high-level causal flow from the input text X, through LLM-perceived concepts $\mathcal{C}$, to the model prediction $\hat{Y}$, while allowing dependencies among concepts. We impose three structural constraints: X has only outgoing edges and serves as the root of the graph;[1] $\hat{Y}$ has only incoming edges, and the effect of the input text on the prediction is mediated by the concept variables. Accordingly, our task is to recover the concept set $\mathcal{C}$ and the edge set E.

3.2 Concept Discovery and Annotation

Label Prediction. We first replace the ground-truth labels in $\mathcal{D}$ with the LLM's predictions $\hat{y} = f(x)$. All downstream phases use these predicted labels, so the recovered graph reflects the model's perspective rather than the dataset labels (Box E.1).

Discriminative Concept Discovery. We extract a set of concepts $\mathcal{C}$ that differentiate among the classes in $\mathcal{Y}$. The procedure processes $\mathcal{D}_{\text{train}}$ in small, class-balanced batches, where each batch B contains $|B|/|\mathcal{Y}|$ instances from each class. For each batch, the model is shown the examples and proposes new candidate concepts, which are added to $\mathcal{C}$ (prompts in Boxes E.2 and E.3). Every ten batches, we filter the accumulated concepts. To do so, we first annotate each example seen so far with a concept vector $\phi(x)$ (prompt in Box E.4), where each concept takes a scalar value from $\mathcal{V}$, corresponding to a subset $S \subseteq \mathcal{Y}$. We then retain only concepts that are both relevant and discriminative. Specifically, a concept is retained if the fraction of examples for which its assigned value maps to a strictly partial subset of classes (i.e., neither $S = \emptyset$ nor $S = \mathcal{Y}$) exceeds the threshold $\tau = 1/|\mathcal{Y}|$.

[1] For simplicity, we visualize graphs without the text node.

Intuitively, this removes concepts that are rarely expressed in the text and therefore provide little evidence for a text-to-concept edge, as well as concepts that are broadly aligned with all classes and therefore provide little evidence for a concept-to-prediction edge. After processing all batches in $\mathcal{D}_{\text{train}}$, we annotate the examples in $\mathcal{D}_{\text{test}}$ with concept vectors and apply the same filtering criterion once more, yielding the final concept set $\mathcal{C}$. Algorithm 1 in the appendix provides the full procedure.

3.3 Data Expansion and Causal Discovery

Causal discovery from observational data requires samples that adequately cover the joint variable space $(\phi(X), \hat{Y})$. Since the mechanism under study is an LLM, we can cheaply query it to generate textual counterfactuals. Our expansion phase, therefore, constructs chains of counterfactual texts, producing concept vectors that more densely populate the relevant regions of the concept space.

Markov Chain in Text Space. Markov Chain Monte Carlo (MCMC) methods provide a principled way to explore complex, high-dimensional spaces by constructing chains whose samples approximate a target distribution (Robert and Casella, 2004; Andrieu et al., 2003). Inspired by this idea, we introduce a targeted data-expansion algorithm that starts from a sparse initial dataset and generates a denser set of examples over the relevant domain. Rather than operating directly in the discrete concept space $\mathcal{V}^n$, the algorithm operates in the raw text space: at each step, the LLM proposes a counterfactual, i.e., a textual perturbation, which is then mapped to a concept vector by ϕ and retained only if it induces the intended conceptual shift. Operating in text space preserves linguistic coherence while allowing us to explore realizable regions of $\mathcal{V}^n$, i.e., concept configurations that can be expressed by natural text.

Each original example $x \in \mathcal{D}_{\text{train}}$ initiates an independent expansion process lasting $K = 11$ steps. At each step, for a current example x, we iterate through all $c_i \in \mathcal{C}$ and sample a target class $y^* \in \mathcal{Y}$. Let $S \subseteq \mathcal{Y}$ be the subset of classes mapped to by the scalar concept value $\phi(x)[c_i]$. We then choose a directional shift $dx \in \{\text{MORE}, \text{LESS}\}$ according to whether y^* is included in this subset: $dx = \text{MORE}$ if $y^* \notin S$, and $dx = \text{LESS}$ otherwise. Thus, the LLM-generated *counterfactual proposal* $\tilde{x}$ either attempts to introduce an alignment between c_i and y^* or to remove an existing one.

For example, if x is *"I feel a strong headache that gets worse in the light"*, the target concept is *SensitivityToLight*, and $y^\star$ is *Migraine*. If the LLM aligns this concept with *Migraine*, the sampled direction is $dx = \text{LESS}$. The LLM is then prompted to rewrite the text to reduce the alignment between *SensitivityToLight* and *Migraine*, while keeping other concepts fixed. A possible proposal $\tilde{x}$ is *"I feel a strong headache since this morning."*.

Acceptance Criteria. A proposal $\tilde{x}$ is appended to $\mathcal{D}_{\text{mcmc}}$ if it satisfies two conditions, akin to a Metropolis–Hastings test (Hastings, 1970): **(i) Target alignment**: We compute $\phi(\tilde{x})$ (concurrently eliciting reasoning for later use). Let $\tilde{S} \subseteq \mathcal{Y}$ be the subset mapped from $\phi(\tilde{x})[c_i]$. We require $y^* \in \tilde{S}$ if $dx = \text{MORE}$, and $y^* \notin \tilde{S}$ otherwise. **(ii) Minimal side effects**: The count of drifted non-target concepts must be bounded by a tolerance $\epsilon \in \{1, 2\}$, allowing for natural causal correlations: $\sum_{c_j \neq c_i} \mathbb{I}[\phi(\tilde{x})[c_j] \neq \phi(x)[c_j]] \leq \epsilon$.

Recursive Refinement. If a proposal fails either acceptance condition, we re-prompt the LLM with feedback indicating which criterion was violated, together with the reasoning produced during the annotation of $\tilde{x}$. For each concept, the chain attempts up to $R = 5$ regenerations. If all attempts fail, the chain proceeds without appending a sample for that concept. In summary, each original example can contribute up to $K|\mathcal{C}|$ counterfactual examples to $\mathcal{D}_{\text{mcmc}}$, though the actual number may be smaller when proposals fail the acceptance criteria after all refinement attempts. The full MCMC procedure is described in Algorithm 2 in Appendix D.

Rationale. Our MCMC-inspired expansion densifies the latent concept space by exploring combinations beyond the original dataset. Starting from each example, the chain iteratively proposes local changes to concept assignments and prompts the LLM to generate corresponding counterfactuals. Proposals are accepted only if the generated example is valid and matches the intended configuration; otherwise, they are rejected to avoid inconsistent or unrealizable regions. Thus, the procedure stochastically explores the valid concept manifold, providing richer support for causal discovery.

Causal Discovery via σ-CG. We apply the σ-CG causal discovery algorithm (Forré and Mooij, 2018) to the expanded, annotated dataset $\mathcal{D}_{\text{mcmc}}$, obtaining a directed graph $\mathcal{G}$ over the variables V. We choose σ-CG because it accommodates cyclic

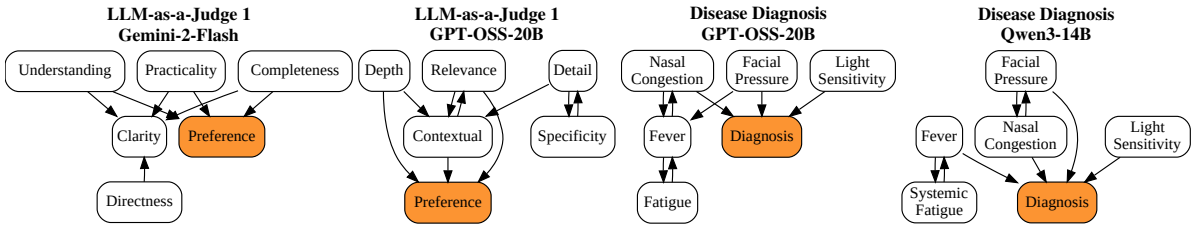


Figure 3: Illustrations of four causal graphs. All causal graphs are provided in AppendixC.

causal structures, which may arise in LLM inference and therefore cannot be ruled out a priori, and because it supports the discrete variables produced by our concept annotations. As background knowledge, we impose that $\hat{y}$ is the unique sink node, and enforce this constraint during edge orientation.

4 Experimental Setup

Models. We evaluate our framework and explain three LLMs: Gemini-2-Flash (Team et al., 2025), Qwen3-14B (Yang et al., 2025), and gpt-OSS-20b (OpenAI, 2025) For the open-weights models (Qwen3-14B and gpt-OSS-20b) we use the vLLM framework (Kwon et al., 2023). For each model, we use two temperature settings: $\mathcal{M}_{cls}$ with $\tau = 0$ for deterministic inference and reproducible labeling, and $\mathcal{M}_{gen}$ with $\tau = 0.5$ for concept extraction and MCMC-based counterfactual generation.

Datasets. We evaluate on three classification tasks, scaling batch sizes by input length and class count. (1) *Disease Diagnosis (DD, $N = 1448$, $B = 9$)*: the LIBERTY dataset, a synthetic medical corpus in which patient descriptions are classified as Migraine, Sinusitis, or Influenza (Toker et al., 2026). (2) *Sentiment Analysis (SA, $N = 2096$, $B = 10$)*: the IMDB dataset, in which movie reviews are classified as Positive or Negative (Maas et al., 2011). (3) *LLM-as-a-Judge (LAJ, $N = 395$, $B = 10$)*: a preference prediction task constructed from Reddit, where each input contains a question and two candidate responses and the model selects the preferred one (Calderon et al., 2025). To mitigate positional bias, each pair is presented twice with the response order swapped. For Disease Diagnosis and Sentiment Analysis, we construct a single causal graph per task, capturing the reasoning structure that the model applies uniformly across all inputs in the domain. LAJ, however, poses a distinct challenge: queries span unrelated topics (e.g., nutrition, programming, travel advice), and the criteria governing the model's preference for one query bear little relation to those for another. A single graph cannot meaningfully capture such heterogeneous reasoning. We therefore adopt a *query-level* approach: for each query, we generate a large set of diverse response pairs, forming a self-contained dataset that captures how the model adjudicates responses *to that specific question*. We then construct a separate causal graph per query, explaining what drives the model's preference within a fixed topical context (see Appendix A.1).

5 Results

5.1 Predictive Fidelity Evaluation

Our goal is to recover a causal graph over the LLM's latent inference process, and there is naturally no ground-truth graph for direct evaluation. We therefore evaluate the learned graphs through *predictive fidelity*: If a graph captures meaningful dependencies in the model's inference, the direct causal parents should screen off indirect variables, making the parents of each node more predictive of that node than other concept subsets.

We test this using a post hoc prediction task. For each target node $v \in \mathcal{C} \cup \{\hat{Y}\}$, we train a multinomial logistic regression model to predict v from a one-hot encoding of its graph-based parent set $Pa(v)$. We compare it to models trained on all possible concept subsets $Z \subseteq \mathcal{C}$ that do not fully contain the parent set, i.e., $Pa(v) \not\subseteq Z$. We evaluate both on the final model output $\hat{Y}$ and the concept nodes $c_i \in \mathcal{C}$, using 10-fold cross-validation (CV).

As shown in Table 1, graph-based predictors outperform the average alternative subset in every setting, across all models, datasets, and target types. This indicates that the discovered parent sets capture information relevant to both final predictions and concept-level dependencies. The Top-3 results further strengthen this conclusion: in every setting but one, the causal-graph parent set ranks among the best predictors in a majority of CV cases, despite being compared against tens of alternatives.

Target	Disease Diagnosis			Sentiment Analysis			LLM-as-a-Judge		
	CG Acc.	Others Acc.	CG in Top-3	CG Acc.	Others Acc.	CG in Top-3	CG Acc.	Others Acc.	CG in Top-3
Gemini-2-Flash									
Prediction $\hat{y}$	**0.67**	0.61	**100.0%**	**0.80**	0.77	**80.0%**	**0.79**	0.74	**57.5%**
Concepts $\mathcal{C}$	**0.54**	0.51	**96.7%**	**0.61**	0.58	**95.0%**	**0.83**	0.72	**80.8%**
GPT-OSS 20B									
Prediction $\hat{y}$	**0.67**	0.59	**100.0%**	**0.84**	0.78	**100.0%**	**0.83**	0.78	**53.0%**
Concepts $\mathcal{C}$	**0.53**	0.48	**76.7%**	**0.60**	0.55	**78.0%**	**0.77**	0.67	**86.2%**
Qwen3-14B									
Prediction $\hat{y}$	**0.63**	0.56	50.0%	**0.80**	0.75	**80.0%**	**0.64**	0.62	**64.4%**
Concepts $\mathcal{C}$	**0.65**	0.54	**92.5%**	**0.61**	0.60	**97.5%**	**0.88**	0.80	**82.9%**
Random Baseline									
Prediction $\hat{y}$	0.33	0.33	≈12%	0.50	0.50	≈12%	0.50	0.50	≈12%
Concepts $\mathcal{C}$	0.12	0.12	≈27%	0.25	0.25	≈27%	0.25	0.25	≈27%

Table 1: **Causal-Graph Prediction Results:** Average 10-fold cross-validation results. In "Prediction $\hat{y}$", logistic regression predicts the task label; in "Concepts $\mathcal{C}$", we train one logistic regression per concept to predict its value and report the mean accuracy across concepts. "CG Acc." uses the causal-graph parent concepts as inputs, "Others Acc." averages over concept combinations that do not contain the full parent set, and "CG in Top-3" is the percentage of cases in which CG Acc. ranks among the top three concept combinations.

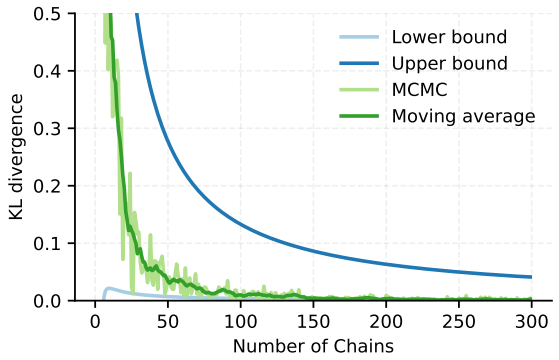


Figure 4: KL-divergence convergence during MCMC-inspired data expansion on Sentiment Analysis for GPT-OSS-20B. The empirical curve is compared against two bounds: perfect overlap (samples fall within the covered regions, lower bound) and orthogonal expansion (samples occupy unseen regions, upper bound).

5.2 Analysis of the Causal Graphs

We next discuss the learned concept sets and causal graphs. On the synthetic Disease Diagnosis dataset, the extracted concepts are consistent across LLMs. As shown in Table 2, they successfully recover the ground-truth variables (e.g., specific symptoms) defined in the causal graph used to generate the dataset (Toker et al., 2026). The learned graphs also reveal that different LLMs can follow different concept-level reasoning patterns. Even when models rely on similar concepts, their causal topologies differ (Figure 3). This model-specific structure is even more pronounced in the naturalistic Sentiment Analysis and LAJ tasks, where the extracted concepts themselves vary across models (See Table 2), suggesting that models rely on different reasoning mechanisms. To summarize, these differences matter for deployment: choosing among models requires understanding not only their performance but also whether their underlying reasoning aligns with stakeholders' expectations and constraints.

5.3 MCMC Convergence and Stability.

Multi-chain diagnostics such as the Gelman-Rubin statistic are not directly applicable in our single-chain-per-instance setting. Instead, we monitor convergence by tracking the empirical probability distribution over the $|\mathcal{V}|^{|\mathcal{C}|}$ combinatorial concept state space. Following each iteration, we calculate the Kullback-Leibler (KL) divergence between the updated global probability vector and the previous one. A naive KL decrease is not sufficient evidence of convergence: as the sample pool grows, each new counterfactual carries diminishing marginal weight, mechanically reducing the KL regardless of whether the chain has genuinely stabilized. To separate genuine convergence from this artifact, we bound the empirical KL between two extremes (Equations 1, 2, in Appendix A.5): an orthogonal expansion upper bound, where every new sample occupies a previously unseen concept state, and a perfect overlap lower bound, where every new sample duplicates an existing one.

A desirable trajectory begins near the upper

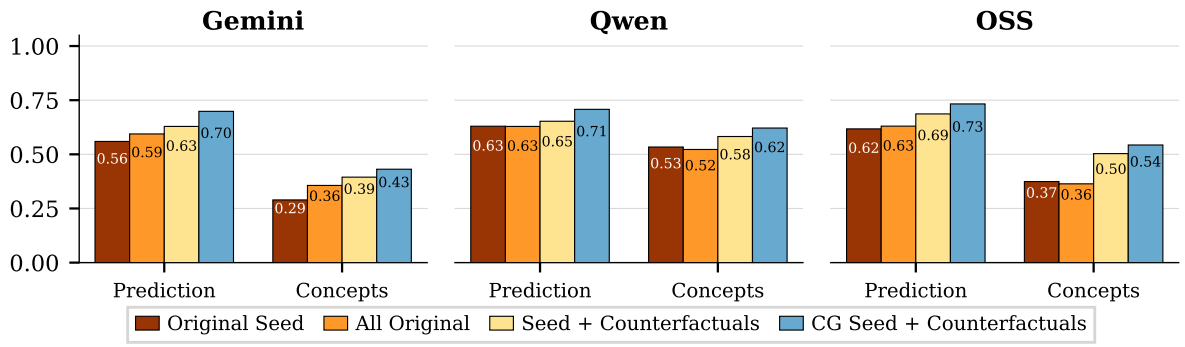


Figure 5: **Impact of MCMC:** Mean accuracy across DD and SA datasets and all concept combinations, under three training regimes: All Original (full original dataset), Original Seed (subset used as a seed for the MCMC), and Seed + Counterfactuals. For each target $v \in \mathcal{C} \cup \{\hat{y}\}$, a logistic regression is trained over all possible concept combinations. The blue bars show the accuracy when the input consists of the parents based on the causal graph.

bound, indicating that the chain is actively discovering new valid concept states, and gradually approaches the lower bound, indicating that the accepted samples increasingly come from a stable, realizable region. As shown in Figure 4, our empirical KL follows precisely this trajectory. Finally, we verify that this distributional stabilization is accompanied by structural stability: once the KL stabilizes, the Hamming distance between causal edge sets recovered across successive iterations drops to zero (Figures 8, 9), suggesting that additional augmentation no longer changes the learned graph.

5.4 Impact of Counterfactual Augmentations

We evaluate the contribution of MCMC expansion by comparing three data regimes: (i) *All Original* (the full original dataset); (ii) *Original Seed* (the subset seeding the MCMC chains); and (iii) *Seed + Counterfactual*. For each regime and each node $v \in \mathcal{C} \cup \{\hat{y}$, we train a multinomial logistic regression predicting v from every possible subset of the remaining concepts. Figure 5 reports prediction accuracy averaged first over all concept subsets, then across SA and DD datasets; we also present the accuracy of the parent subset identified by the causal graph. Note that, unlike in §5.1, this time we use a different held-out test set, and the average accuracy over combinations includes the parent subset itself and all of its supersets.

Two patterns emerge. First, the counterfactuals achieve the highest predictive accuracy for both concept nodes and prediction across all models. The MCMC-generated counterfactuals introduce concept-state combinations that are absent in the original data, enabling more accurate estimation of the dependencies between concepts. Second, consistent with §5.1, the causal graph's parent subsets outperform the average over all concept subsets. Together, these results demonstrate the necessity of our counterfactual augmentation stage.

6 Conclusion

In this paper, we propose a novel causal framework for LLM explainability that models the inference process with a causal graph over human-interpretable concepts. We believe this representation can provide a more accessible and actionable form of interpretability for both model developers and domain experts using LLMs in practice. A central technical contribution of this work is our MCMC-inspired data expansion algorithm, designed to address the sparsity of observational data in latent concept spaces. By iteratively generating counterfactuals, the method produces denser and more representative datasets that facilitate accurate and stable causal discovery. Our experiments further address the challenge of evaluation in the absence of gold-standard reasoning graphs by introducing predictive and structural validation protocols that consistently demonstrate both the quality of the discovered causal graphs and the contribution of the MCMC-inspired expansion process.

More broadly, we hope this work will contribute to a new generation of process-oriented explainability methods for LLMs, focused on modeling inference mechanisms rather than explaining isolated predictions. In future work, we plan to extend the framework to additional languages, open-ended generation settings, and multimodal domains, while further broadening the empirical evaluation to move toward more standardized and stakeholder-oriented explainability frameworks.

7 Limitations

Sensitivity to Batch Pairing in Concept Extraction. Concept extraction is performed in small batches due to LLM output token limits, and we run the process only once due to inference costs. As a result, the candidate concepts may depend on the arbitrary grouping of examples within each batch. Different batch assignments could highlight distinct contrastive features and yield a different or more comprehensive set of concepts $\mathcal{C}$. Future work could mitigate this with multiple shuffled extraction passes when computational budget allows.

Restricted Evaluation to Local Parents. Our framework is designed to recover a global concept-level causal graph, but the current validation protocol primarily evaluates local predictive fidelity through discovered parent sets. This demonstrates that the immediate predictors of each node are informative and structurally stable, but it does not directly verify longer multi-hop causal chains or the broader hierarchy of the recovered graph.

Dependence on Self-Annotation and Generation. During counterfactual generation, we rely on the target LLM to annotate concepts, generate counterfactuals, and judge whether proposed edits satisfy the intended constraints. Because LLM-generated reasoning and self-assessments are not guaranteed to be faithful, errors in these intermediate steps may propagate. Although our acceptance criteria reduce this risk by checking for target alignment and concept drift, future work should investigate external validation, human audits, or multi-model agreement to further assess the quality of annotations and counterfactuals.

References

Chirag Agarwal, Sree Harsha Tanneru, and Himabindu Lakkaraju. 2024. Faithfulness vs. plausibility: On the (un)reliability of explanations from large language models. *CoRR*, abs/2402.04614.

Constantin F. Aliferis, Alexander R. Statnikov, Ioannis Tsamardinos, Subramani Mani, and Xenofon D. Koutsoukos. 2010. Local causal and markov blanket induction for causal discovery and feature selection for classification part I: algorithms and empirical evaluation. *J. Mach. Learn. Res.*, 11:171–234.

Christophe Andrieu, Nando de Freitas, Arnaud Doucet, and Michael I. Jordan. 2003. An introduction to MCMC for machine learning. *Mach. Learn.*, 50(1-2):5–43.

Stephan Bongers, Patrick Forré, Jonas Peters, and Joris M. Mooij. 2021. Foundations of structural causal models with cycles and latent variables. *The Annals of Statistics*, 49(5).

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, and 12 others. 2020. Language models are few-shot learners. *CoRR*, abs/2005.14165.

Sébastien Bubeck, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrke, Eric Horvitz, Ece Kamar, Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott Lundberg, Harsha Nori, Hamid Palangi, Marco Tulio Ribeiro, and Yi Zhang. 2023. Sparks of artificial general intelligence: Early experiments with gpt-4. *Preprint*, arXiv:2303.12712.

Nitay Calderon, Liat Ein-Dor, and Roi Reichart. 2025. Multi-domain explainability of preferences. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing, EMNLP 2025, Suzhou, China, November 4-9, 2025*, pages 14542–14575. Association for Computational Linguistics.

Nitay Calderon and Roi Reichart. 2025. On behalf of the stakeholders: Trends in NLP model interpretability in the era of LLMs. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 656–693, Albuquerque, New Mexico. Association for Computational Linguistics.

David Maxwell Chickering. 2002. Optimal structure identification with greedy search. *J. Mach. Learn. Res.*, 3:507–554.

Matthew Dahl, Varun Magesh, Mirac Suzgun, and Daniel E Ho. 2024. Large legal fictions: Profiling legal hallucinations in large language models. *Journal of Legal Analysis*, 16(1):64–93.

Amir Feder, Katherine A. Keith, Emaad Manzoor, Reid Pryzant, Dhanya Sridhar, Zach Wood-Doughty, Jacob Eisenstein, Justin Grimmer, Roi Reichart, Margaret E. Roberts, Brandon M. Stewart, Victor Veitch, and Diyi Yang. 2022. Causal inference in natural language processing: Estimation, prediction, interpretation and beyond. *Trans. Assoc. Comput. Linguistics*, 10:1138–1158.

Amir Feder, Nadav Oved, Uri Shalit, and Roi Reichart. 2021. Causalm: Causal model explanation through counterfactual language models. *Comput. Linguistics*, 47(2):333–386.

Tao Feng, Lizhen Qu, Niket Tandon, Zhuang Li, Xiaoxi Kang, and Gholamreza Haffari. 2025. On the reliability of large language models for causal discovery. In *Proceedings of the 63rd Annual Meeting of the*

Association for Computational Linguistics (Volume 1: Long Papers), pages 9565–9590, Vienna, Austria. Association for Computational Linguistics.

Patrick Forré and Joris M Mooij. 2018. Constraint-based causal discovery for non-linear structural causal models with cycles and latent confounders. *arXiv preprint arXiv:1807.03024*.

Yair Ori Gat, Nitay Calderon, Amir Feder, Alexander Chapanin, Amit Sharma, and Roi Reichart. 2024. Faithful explanations of black-box NLP models using llm-generated counterfactuals. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

Andrew Gelman and Donald B. Rubin. 1992. Inference from Iterative Simulation Using Multiple Sequences. *Statistical Science*, 7(4):457 – 472.

Clark Glymour, Kun Zhang, and Peter Spirtes. 2019. Review of causal discovery methods based on graphical models. *Frontiers in Genetics*, Volume 10 - 2019.

W. K. Hastings. 1970. Monte carlo sampling methods using markov chains and their applications. *Biometrika*, 57:97–109.

Sarthak Jain and Byron C. Wallace. 2019. Attention is not explanation. *CoRR*, abs/1902.10186.

Been Kim, Martin Wattenberg, Justin Gilmer, Carrie J. Cai, James Wexler, Fernanda B. Viégas, and Rory Sayres. 2018. Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (TCAV). In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, Proceedings of Machine Learning Research, pages 2673–2682. PMLR.

Hassane Kissane, Achim Schilling, and Patrick Krauss. 2025. Probing internal representations of multi-word verbs in large language models. *CoRR*, abs/2502.04789.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. *Preprint*, arXiv:2309.06180.

Tian Lan, Jinyuan Xu, Xue He, Jenq-Neng Hwang, and Lei Li. 2025. Attention consistency for llms explanation. In *Findings of the Association for Computational Linguistics: EMNLP 2025, Suzhou, China, November 4-9, 2025*, pages 1736–1750. Association for Computational Linguistics.

Tamera Lanham, Anna Chen, Ansh Radhakrishnan, Benoit Steiner, Carson Denison, Danny Hernandez, Dustin Li, Esin Durmus, Evan Hubinger, Jackson Kernion, Kamile Lukosiute, Karina Nguyen, Newton Cheng, Nicholas Joseph, Nicholas Schiefer, Oliver Rausch, Robin Larson, Sam McCandlish, Sandipan Kundu, and 11 others. 2023. Measuring faithfulness in chain-of-thought reasoning. *CoRR*, abs/2307.13702.

Peiwen Li, Xin Wang, Zeyang Zhang, Yuan Meng, Fang Shen, Yue Li, Jialong Wang, Yang Li, and Wenwu Zhu. 2024. Realtcd: Temporal causal discovery from interventional data with large language model. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*, CIKM '24, page 4669–4677, New York, NY, USA. Association for Computing Machinery.

Chenxi Liu, Yongqiang Chen, Tongliang Liu, Mingming Gong, James Cheng, Bo Han, and Kun Zhang. 2024. Discovery of the hidden world with large language models. In *Advances in Neural Information Processing Systems*, volume 37, pages 102307–102365. Curran Associates, Inc.

Qing Lyu, Shreya Havaldar, Adam Stein, Li Zhang, Delip Rao, Eric Wong, Marianna Apidianaki, and Chris Callison-Burch. 2023. Faithful chain-of-thought reasoning. In *Proceedings of the 13th International Joint Conference on Natural Language Processing and the 3rd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics, IJCNLP 2023 -Volume 1: Long Papers, Nusa Dua, Bali, November 1 - 4, 2023*, pages 305–329. Association for Computational Linguistics.

Jing Ma. 2025. Causal inference with large language model: A survey. In *Findings of the Association for Computational Linguistics: NAACL 2025, Albuquerque, New Mexico, USA, April 29 - May 4, 2025*, Findings of ACL, pages 5886–5898. Association for Computational Linguistics.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA. Association for Computational Linguistics.

Aaron Mueller, Jannik Brinkmann, Millicent L. Li, Samuel Marks, Koyena Pal, Nikhil Prakash, Can Rager, Aruna Sankaranarayanan, Arnab Sen Sharma, Jiuding Sun, Eric Todd, David Bau, and Yonatan Belinkov. 2024. The quest for the right mediator: A history, survey, and theoretical grounding of causal interpretability. *CoRR*, abs/2408.01416.

Jesutofunmi A Omiye, Haiwen Gui, Shawheen J Rezaei, James Zou, and Roxana Daneshjou. 2024. Large language models in medicine: the potentials and pitfalls: a narrative review. *Annals of internal medicine*, 177(2):210–220.

OpenAI. 2025. gpt-oss-120b & gpt-oss-20b model card. *Preprint*, arXiv:2508.10925.

Debjit Paul, Robert West, Antoine Bosselut, and Boi Faltings. 2024. Making reasoning matter: Measuring and improving faithfulness of chain-of-thought reasoning. In *Findings of the Association for Computational Linguistics: EMNLP 2024, Miami, Florida, USA, November 12-16, 2024*, Findings of ACL, pages 15012–15032. Association for Computational Linguistics.

Judea Pearl. 2000. *Causality: Models, Reasoning, and Inference*. Cambridge University Press.

Luyu Qiu, Yi Yang, Caleb Chen Cao, Jing Liu, Yueyuan Zheng, Hilary Hei Ting Ngai, Janet H. Hsiao, and Lei Chen. 2021. Resisting out-of-distribution data problem in perturbation of XAI. *CoRR*, abs/2107.14000.

Christian P. Robert and George Casella. 2004. *Monte Carlo Statistical Methods*. Springer Texts in Statistics. Springer.

Manish Sanwal. 2025. Layered chain-of-thought prompting for multi-agent LLM systems: A comprehensive approach to explainable large language models. *CoRR*, abs/2501.18645.

Kartik Sharma, Yiqiao Jin, Rakshit S. Trivedi, and Srijan Kumar. 2025. Efficient knowledge probing of large language models by adapting pre-trained embeddings. *CoRR*, abs/2508.06030.

Shriyank Somvanshi, Md Monzurul Islam, Amir Rafe, Anannya Ghosh Tusti, Arka Chakraborty, Anika Baitullah, Tausif Islam Chowdhury, Nawaf Alnawmasi, Anandi Dutta, and Subasish Das. 2026. Bridging the black box: A survey on mechanistic interpretability in ai. *ACM Comput. Surv.*, 58(8).

Peter Spirtes, Clark N Glymour, and Richard Scheines. 2000. *Causation, prediction, and search*. MIT press.

Gemini Team, Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M. Dai, Anja Hauth, Katie Millican, David Silver, Melvin Johnson, Ioannis Antonoglou, Julian Schrittwieser, Amelia Glaese, Jilin Chen, Emily Pitler, Timothy Lillicrap, Angeliki Lazaridou, and 1332 others. 2025. Gemini: A family of highly capable multimodal models. *Preprint*, arXiv:2312.11805.

Gilat Toker, Nitay Calderon, Ohad Amosy, and Roi Reichart. 2026. Liberty: A causal framework for benchmarking concept-based explanations of llms with structural counterfactuals. *Preprint*, arXiv:2601.10700.

Miles Turpin, Julian Michael, Ethan Perez, and Samuel R. Bowman. 2023. Language models don't always say what they think: Unfaithful explanations in chain-of-thought prompting. *Preprint*, arXiv:2305.04388.

Kevin Ro Wang, Alexandre Variengien, Arthur Conmy, Buck Shlegeris, and Jacob Steinhardt. 2023. Interpretability in the wild: a circuit for indirect object identification in GPT-2 small. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net.

Tongshuang Wu, Marco Túlio Ribeiro, Jeffrey Heer, and Daniel S. Weld. 2021. Polyjuice: Automated, general-purpose counterfactual generation. *CoRR*, abs/2101.00288.

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, and 41 others. 2025. Qwen3 technical report. *Preprint*, arXiv:2505.09388.

Jingyuan Yang, Dapeng Chen, Yajing Sun, Rongjun Li, Zhiyong Feng, and Wei Peng. 2024. Enhancing semantic consistency of large language models through model editing: An interpretability-oriented approach. In *Findings of the Association for Computational Linguistics, ACL 2024, Bangkok, Thailand and virtual meeting, August 11-16, 2024*, Findings of ACL, pages 3343–3353. Association for Computational Linguistics.

Catherine Yeh, Yida Chen, Aoyu Wu, Cynthia Chen, Fernanda B. Viégas, and Martin Wattenberg. 2024. Attentionviz: A global view of transformer attention. *IEEE Trans. Vis. Comput. Graph.*, 30(1):262–272.

Yue Yu, Jie Chen, Tian Gao, and Mo Yu. 2019. Dag-gnn: Dag structure learning with graph neural networks. In *International conference on machine learning*, pages 7154–7163. PMLR.

Alessio Zanga, Elif Ozkirimli, and Fabio Stella. 2022. A survey on causal discovery: Theory and practice. *International Journal of Approximate Reasoning*, 151:101–129.

Matej Zečević, Moritz Willig, Devendra Singh Dhami, and Kristian Kersting. 2023. Causal parrots: Large language models may talk causality but are not causal. *arXiv preprint arXiv:2308.13067*.

Hanyu Zhang, Xiting Wang, Chengao Li, Xiang Ao, and Qing He. 2025. Controlling large language models through concept activation vectors. In *Thirty-Ninth AAAI Conference on Artificial Intelligence, Thirty-Seventh Conference on Innovative Applications of Artificial Intelligence, Fifteenth Symposium on Educational Advances in Artificial Intelligence, AAAI 2025, Philadelphia, PA, USA, February 25 - March 4, 2025*, pages 25851–25859. AAAI Press.

Muru Zhang, Ofir Press, William Merrill, Alisa Liu, and Noah A. Smith. 2024. How language model hallucinations can snowball. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*, Proceedings of Machine Learning Research, pages 59670–59684. PMLR / OpenReview.net.

Haiyan Zhao, Hanjie Chen, Fan Yang, Ninghao Liu, Huiqi Deng, Hengyi Cai, Shuaiqiang Wang, Dawei Yin, and Mengnan Du. 2023. Explainability for large language models: A survey. *Preprint*, arXiv:2309.01029.

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

Yu Zheng, Yuan Yuan, Yong Li, and Paolo Santi. 2025. Probing neural topology of large language models. *CoRR*, abs/2506.01042.

A Supplementary Results and Extended Discussion

A.1 Datasets

Dataset Scale and Subsampling. Our evaluation spans three core classification tasks: Disease Diagnosis (DD), Sentiment Analysis (SA), and LLM-as-a-Judge (LAJ). For the Sentiment Analysis task, we leverage the IMDB movie review corpus (Maas et al., 2011). Given the computational and API cost constraints of running LLM-based pipelines over large corpora, we subsample the IMDB corpus to a representative subset of $N = 2096$ instances.

Feature Isolation and Textual Extraction. We retain only the core textual content relevant to each classification task, discarding all metadata and structural fields. For Disease Diagnosis, we use raw patient descriptions from the LIBERTY benchmark's synthetic medical corpus (Toker et al., 2026) ($N = 1448$), predicting among Migraine, Sinusitis, and Influenza. For Sentiment Analysis, we use raw movie review texts from IMDB ($N = 2096$), classified as Positive or Negative. For LLM-as-a-Judge, we retain the primary user query and two candidate responses sourced from Reddit (Calderon et al., 2025) ($N = 395$).

Handling Positional Bias in LAJ The LAJ task is susceptible to positional bias: we observed that swapping the order of candidate responses altered the model's classification outcome in over 30% of instances. To mitigate this, we present each pair twice with the response order inverted during both the inference and labeling alignment phase and the data expansion phase. Concretely, for each dataset instance we run two independent MCMC chains in parallel, one per response ordering, ensuring the expanded counterfactual dataset is balanced across both permutations.

A.2 Concept Extraction

Table 2 details the differentiative concepts extracted for every combination of model and dataset. Notably, our results indicate that within the synthetic DD benchmark, the extracted concepts are remarkably similar across different models and faithfully reconstruct the base graph utilized for data generation (Toker et al., 2026). Conversely, the concepts derived from the natural datasets display a higher degree of model-specificity, highlighting how individual models leverage distinct internal representations for the same task.

A.3 Data Expansion

When evaluating the LAJ dataset, our framework constructs a distinct causal graph for each individual query. Consequently, the initial dataset $\mathcal{D}$ for each graph effectively consists of only a single seed example. In our evaluation, we analyze a subset of queries for each model, resulting in $Q = 48$ unique graphs for Gemini, $Q = 18$ for QWEN, and $Q = 10$ for GPT-OSS. Under these conditions, the observed distribution of concepts is fundamentally insufficient for reliable causal discovery. To mitigate this cold-start problem and artificially inflate the data, we apply a "coarse expansion" phase as a mandatory first step. The objective of this phase is to enrich the dataset by guiding the target LLM f to express the candidate concepts across a variety of classification targets.

Let $\mathcal{Y}$ denote the set of all possible class labels. For each initial data instance x and each candidate concept $c_i \in \mathcal{C}$, we first evaluate its current alignment context, denoted as $\phi(x)[c_i] \in \mathcal{V}$. The specific value of this alignment dictates the direction of our textual perturbations. If the concept c_i is currently aligned with a single, specific class $y \in \mathcal{Y}$ (i.e., $\phi(x)[c_i] \mapsto \{y\}$), we aim to observe how the concept behaves under different target labels. Thus, we prompt the model f to rewrite the original text x into new variations, specifically directing it to perturb the instance toward all remaining classes in the complement set $\mathcal{Y} \setminus \{y\}$. Conversely, if the concept is neutral and currently appears identically across all classes (i.e., it is aligned with the full set, $\phi(x)[c_i] \mapsto \mathcal{Y}$), we lack information about its discriminative boundaries. In this case, the expansion is executed across the entire label space, and the model is prompted to generate distinct variations targeted at every individual class in Y.

This perturbation strategy is deliberately exhaus-

Dataset	Model	Extracted Concepts
Sentiment Analysis	Qwen3-14B	Enjoyment, Audience Appeal, Expectation Management, Emotional Impact, Performance Quality
	Gemini-2-Flash	Enjoyment, Fulfilment, Execution, Sincerity, Focus
	gpt-OSS-20b	Recommendation, Enjoyment, Tone, AudienceEngagement, OverallQuality
Disease Diagnosis	Qwen3-14B	Fever, Systemic Fatigue, Facial Pressure, Nasal Congestion, Light Sensitivity
	Gemini-2-Flash	Fever, Facial Pressure, Nasal Congestion, Light Sensitivity
	gpt-OSS-20b	Fever, Fatigue, Facial Pressure, Nasal Congestion, Light Sensitivity
LAJ (Food)	Qwen3-14B	Usefulness and Practicality, Clarity and Specificity, Relevance and Focus, Tone and Professionalism, Feasibility and Realism
	Gemini-2-Flash	Directness, Practicality, Completeness, Clarity, Understanding
	gpt-OSS-20b	Relevance, Depth, Specificity, Contextual, Detail

Table 2: Extracted concepts across different models and datasets. Each set of concepts represents the latent features identified by the specific model as most differentiative for the given task.

tive. At this preliminary stage, we merely instruct the LLM to shift the context toward alternative labels without enforcing any strict downstream constraints on the generation process (the exact prompt template utilized for this step is detailed in Box E.5). By generating these counterfactual examples, we create a denser joint distribution over ϕ, which serve is the starting point of the MCMC expansion phase

A.4 Validation Details

Fidelity and Predictive Validation. To evaluate the predictive necessity and fidelity of the discovered causal structures, we implement an internal validation protocol strictly within the MCMC expended training set.

For each benchmark, we perform 10 rounds of cross-validation on the dataset. In each validation cycle, the data is split into an 80% internal training subset and a 20% held-out validation slice. On the 80% training subset, the σ-CG algorithm is executed to construct the representative causal graph. For each target node $v \in V$ (where $V = \mathcal{C} \cup \{\hat{y}\}$), once the topology is established on the 80% split, we parameterize the functional mechanisms governing the SCM via multinomial logistic regression using its causal parent set, denoted as $Pa(v)$. After parameterization, the predictive capability of the isolated parent sets is evaluated on the remaining 20% held-out validation set. To confirm that the discovered parents $Pa(v)$ constitute the optimal predictive features for each respective node, their classification accuracy is systematically benchmarked against alternative predictor combinations $S \subseteq V \setminus \{v\}$. To ensure a rigorous and fair comparison, we independently train a distinct logistic regression model for each alternative subset Z using the same 80% training split, enforcing the constraint that the full parent set is never encapsulated within the baseline ($Pa(v) \not\subseteq Z$).

Our empirical evaluation demonstrates that across the vast majority of datasets and model architectures, the predictors trained exclusively on the causal parent sets achieve the highest classification accuracy. In a few isolated cases, alternative feature combinations achieve parity (ties) with the causal parents, but never outperform them (see figure 6 in appendix 6.

MCMC Importance. To systematically validate the effectiveness of our MCMC-based data expansion process, we evaluate its impact on the downstream causal discovery across all presented datasets. Specifically, for each individual dataset, we execute the causal discovery process to extract and compare three distinct causal graphs, corresponding to three different training configurations. The first is the *Seed* configuration, which serves as our baseline graph constructed utilizing the full original training set $\mathcal{D}_{\text{train}}$. The second is the *MCMC+Filtered Seed* configuration; this represents our primary graph, extracted by combining the expanded dataset $\mathcal{D}_{\text{mcmc}}$ specifically with the subset of seed instances that successfully initiated valid counterfactual chains. The third is the *Filtered Seed* configuration, which serves as a control

baseline graph learned using only those exact same filtered seed instances, but without the addition of the MCMC-generated counterfactuals.

A key challenge in evaluating causal graphs over sparse observational data is that the initial evaluation set rarely spans the combinatorial concept space adequately. Evaluating model performance on a non-spanning test set can yield skewed or unrepresentative structural metrics. To address this limitation, we leverage our MCMC algorithm to expand the test set as well, forcing a dense coverage of alternative concept configurations. Due to computational constraints and design requirements specifically because the LAJ dataset constructs an isolated graph per unique query this test-set expansion procedure was applied exclusively to the LIBERTY and IMDB datasets.

Following the construction of these three graph configurations, we formalize their structural equations by estimating the potential functions via multiclass logistic regression. Specifically, for each target variable $v \in V$ (where $V = \mathcal{C} \cup \{\hat{y}\}$), we fit a separate logistic regression model to predict its state based exclusively on its set of direct parents in the recovered graph, denoted as $Pa(v)$. To comprehensively evaluate the predictive optimality of this causal structure, we independently train a distinct logistic regression model for every possible alternative subset of predictor concepts $Z \subseteq V \setminus \{v\}$. We then evaluate all learned models on the expanded test set, computing and comparing the predictability accuracy of each variable across the entire combinatorial space. Within the evaluation figures, the performance of the explicitly learned $Pa(v)$ configuration is highlighted with a star symbol.

The empirical results demonstrate the overall value of the MCMC expansion phase (see Figure 7). While the parent-conditioned accuracy derived from the *MCMC+Filtered Seed* graph consistently outperforms both baseline configurations (*Seed* and *Filtered Seed*) for the majority of concepts, we do observe a few isolated exceptions across the evaluated settings. The most prominent deviation occurs with the Gemini model evaluated on the LIBERTY dataset, where the structural accuracy for certain intermediate concepts exhibits suboptimal performance under the expanded regime. Importantly, this degradation remains strictly localized; for the primary classification target (i.e., the *Diagnosis* concept), the *MCMC+Filtered Seed* regime actually improves predictive accuracy compared to the two baseline training configurations. While minor variations exist across nodes, the specifically learned graph-based parent configuration consistently achieves top-tier accuracy within each training regimes, decisively outperforming the vast majority of alternative concept combinations. Ultimately, in the majority of instances, the "center of mass" of the predictive accuracy distribution across the combinatorial space exhibits a clear upward shift under the *MCMC+Filtered Seed* regime. This robust general tendency confirms that the MCMC algorithm successfully expands the observational data distribution while preserving the underlying structural causal signal.

A.5 MCMC Convergence and Stability

Limitations of Classic Diagnostics. Classic MCMC convergence metrics, such as the Gelman-Rubin statistic (Gelman and Rubin, 1992), require knowing the target probability distribution and computing variances across multiple independent chains. Our setting lacks both. First, the underlying probability distribution of our latent concepts is unknown. Second, although classic methods also initialize from multiple starting points, they require running several independent chains from each point to measure variance. In contrast, we run only a single chain per text example. Consequently, we cannot compute traditional variance-based diagnostics, necessitating our tailored KL-based convergence metric.

State Space and Probability Vector. To measure convergence, we define the state space of our extracted concepts. Given a set of concepts $\mathcal{C}$, where each concept can take one of m possible labels, the total number of unique concept combinations is $m^{|\mathcal{C}|}$. We flatten this combinatorial space into a single global probability vector of size $m^{|\mathcal{C}|}$. Each index corresponds to a specific combination of concept labels, and the value at that index represents its empirical probability in our generated data. After each MCMC iteration, we update this vector and calculate the KL divergence from the previous state (see example in B).

Theoretical Bounding Scenarios. Because the global probability vector is updated cumulatively, the relative impact of each new iteration naturally decreases, mathematically forcing the KL divergence to shrink over time. To determine whether the drop in KL divergence indicates genuine convergence rather than this structural artifact, we established two theoretical boundaries under the uni-

form distribution assumption. Let h denote the accumulated number of instances evaluated up to the current stage, and s denote the number of newly added counterfactual instances in the current iteration:

- **Convergence Bound (Perfect Overlap):** Occurs when newly generated samples distribute proportionally across the already populated bins, perfectly mirroring the existing empirical distribution. the new samples act as a representative sub-distribution of the previous step. Since the finite number of samples added per iteration cannot cover the entire target space, this proportional overlap implies the algorithm has mapped the relevant support and stabilized. The closed-form KL divergence for this bound is given by:

$$KL_{\text{overlap}} = \frac{2s}{h+s} \log\left(\frac{2h}{h+s}\right) + \frac{h}{h+s} \log\left(\frac{h}{h+s}\right) \quad (1)$$

- **Non-Convergence Bound (Orthogonal Expansion):** Occurs when new samples fall into completely empty bins. This demonstrates that the algorithm is still exploring entirely new regions, meaning the underlying distribution has not yet stabilized. To prevent undefined logarithmic evaluations for previously empty bins, we introduce a smoothing constant $\epsilon = 10^{-10}$. The closed-form KL divergence for this expansion is given by:

$$KL_{\text{orthogonal}} = \frac{s}{h+s} \log\left(\frac{1}{\epsilon(h+s)}\right) + \frac{h}{h+s} \log\left(\frac{h}{h+s}\right) \quad (2)$$

By plotting the empirical KL divergence against these closed-form boundary curves, we can accurately determine when the state space exploration has concluded.

Convergence Result In this section, we provide the complete empirical evaluation of our MCMC convergence diagnostics across all combinations of target Large Language Models and datasets, extending the representative examples illustrated in the main text. As detailed in Section 5.3, our framework tracks the empirical probability distribution over the combinatorial concept space $\mathcal{C}$ across iterations, evaluating the Kullback-Leibler (KL) divergence against theoretical best-case (perfect overlap) and worst-case (orthogonal exploration) boundaries.

Across all examined models and datasets, the empirical KL divergence curves consistently demonstrate robust and stable convergence (see figure 8 in appendix), systematically shifting away from the upper exploration bound and adhering closely to the lower optimal boundary. This universal trend confirms that the proposed MCMC data expansion successfully convergence prior to the termination of the chains.

However, a distinct structural variation emerges when comparing the architectural paradigms of the datasets. For the IMDB and LIBERTY benchmarks, the global probability distribution is computed across a singular, unified concept space mapped over the entire dataset, resulting in a single empirical convergence curve per model. Conversely, the LAJ framework operates under a distinct constraint, constructing a localized causal graph tailored to each individual query.

Consequently, the combinatorial state space for any single query in LAJ is inherently more constrained and tightly bounded than the global space evaluated in the other datasets. This severe reduction in space complexity manifests empirically as an accelerated rate of convergence, where the chains stabilize significantly faster.

This architectural difference directly dictates the visual representation of the convergence plots. While the IMDB and LIBERTY datasets yield a single empirical convergence curve corresponding to their singular global causal graph, the LAJ dataset evaluation displays multiple empirical curves (depicted as individual green trajectories) within the same plot. Each green trajectory corresponds to the MCMC convergence process of a distinct, query-specific causal graph. These visualizations confirm that despite the independent nature of each query's causal mechanism, every isolated chain strictly satisfies our theoretical convergence criteria.

Stability Results To provide empirical validation for the convergence of our MCMC data expansion algorithm, we conducted a structural consistency check across progressive stages of the expansion process. Since the LAJ dataset yields an independent causal graph for each individual query, this analysis is performed exclusively on the IMDB and LIBERTY benchmarks. For each dataset, we

learn the underlying causal graphs using an increasing number of MCMC iterations, corresponding to a growing volume of expanded counterfactual instances. Specifically, at each evaluated milestone of MCMC iterations, we implement a 10-run cross-validation protocol; in each run, the causal graph is independently learned using a random 80% subset of the data generated up to that stage. The final consensus graph topology for that specific milestone is then determined via a majority vote, such that an edge is included in the final graph if and only if it appeared in 5 or more of the 10 generated graphs. We then quantify the structural distance between the resulting consensus graphs across different expansion stages using the Structural Hamming Distance (SHD), which measures the exact number of edge discrepancies between two graphs (where a distance of 0 denotes topological identity). Our empirical results reveal nuanced convergence patterns across the evaluated models: for Gemini and Qwen, the SHD between subsequent graph configurations consistently diminishes and eventually reaches exactly zero, demonstrating absolute structural invariance. For GPT-OSS, although the structural distances significantly decrease, they stabilize at a low, non-zero baseline rather than completely vanishing (see Figure 9 in the appendix).

B Running Example: Full Pipeline Walkthrough

To provide a concrete understanding of our methodology, this section walks through the four-phase pipeline using a synthetic toy task: classifying the tastiness of a papaya from a short textual description. Let the set of task classes be $\mathcal{Y} = \{\textit{tasty}, \textit{not-tasty}\}$. Each concept takes values in $\mathcal{V} = 2^{\mathcal{Y}}$: $\emptyset$ means that the concept is absent, $\mathcal{Y}$ means that it is aligned with both classes, and singleton sets indicate class-specific alignment.

Phase 1: Label Prediction. Suppose the initial dataset $\mathcal{D}$ contains a seed text $x^{(0)}$: *"Today I ate a bright orange, mushy papaya."* The ground-truth label in the dataset may be *tasty*, but the target LLM predicts $\hat{y} = f(x^{(0)}) = \textit{not-tasty}$, likely because it associates "mushy" with being overripe. We therefore replace the ground-truth label with the LLM prediction, so all downstream stages reflect the model's perspective.

Phase 2: Discriminative Concept Discovery and Representation. We process $\mathcal{D}_{\text{train}}$ in small, class-balanced batches. Given texts and the predicted labels from Phase 1, the LLM proposes candidate concepts that distinguish between the predicted classes. Suppose it proposes *Softness*, *Color*, and *Origin*. Every ten batches, the accumulated candidates are filtered by first annotating the examples seen so far with concept vectors $\phi(x)$.

Concepts are retained only if they are both relevant and discriminative in at least a fraction $\tau = 1/|\mathcal{Y}|$ of the annotated examples. In this binary task, $\tau = 0.5$. Relevant means that the concept is not aligned with $\emptyset$, and discriminative means that it is not aligned with the full class set $\mathcal{Y}$. If *Origin* is annotated as $\emptyset$ for most texts, it fails the relevance criterion. If a concept is usually aligned with $\mathcal{Y}$, it fails the discriminativeness criterion. Suppose *Softness* and *Color* pass the filter; the resulting concept set is $\mathcal{C} = \{c_1 : \textit{Softness}, c_2 : \textit{Color}\}$.

The same annotation step represents each example as a concept vector. For $x^{(0)}$, the word "mushy" aligns *Softness* with *not-tasty*, so $\phi(x^{(0)})[c_1] = \{\textit{not-tasty}\}$. The phrase "bright orange" aligns *Color* with *tasty*, so $\phi(x^{(0)})[c_2] = \{\textit{tasty}\}$. Thus,

$$\phi(x^{(0)}) = \big[\{\textit{not-tasty}\}, \{\textit{tasty}\}\big].$$

Phase 3: MCMC-Inspired Data Expansion. We treat $x^{(0)}$ as the starting point of an independent Markov chain of $K = 11$ steps. Let us trace one transition, from $k = 0$ to $k = 1$:

- **Target selection:** We focus on c_1 (*Softness*), whose current value is $\{\textit{not-tasty}\}$. We uniformly sample a target class $y^* = \textit{tasty}$.
- **Directional shift (dx):** Since $y^* \notin \phi(x^{(0)})[c_1]$, we set $dx = \textsc{More}$, meaning that the proposal should introduce an alignment between *Softness* and *tasty*.
- **Proposal generation:** We prompt the LLM to rewrite $x^{(0)}$ by applying dx to c_1 while keeping c_2 fixed. The LLM generates the counterfactual proposal $\tilde{x}$: *"Today I ate a bright orange, firm papaya."*
- **Acceptance Test: (i) Target alignment:** The LLM re-annotates $\tilde{x}$ and reasons that "firm" indicates good texture. Because $dx = \textsc{More}$, the target condition is $y^* \in \phi(\tilde{x})[c_1]$. If the model assigns $\phi(\tilde{x})[c_1] = \{\textit{tasty}\}$, the target-alignment test passes. **(ii) Minimal**

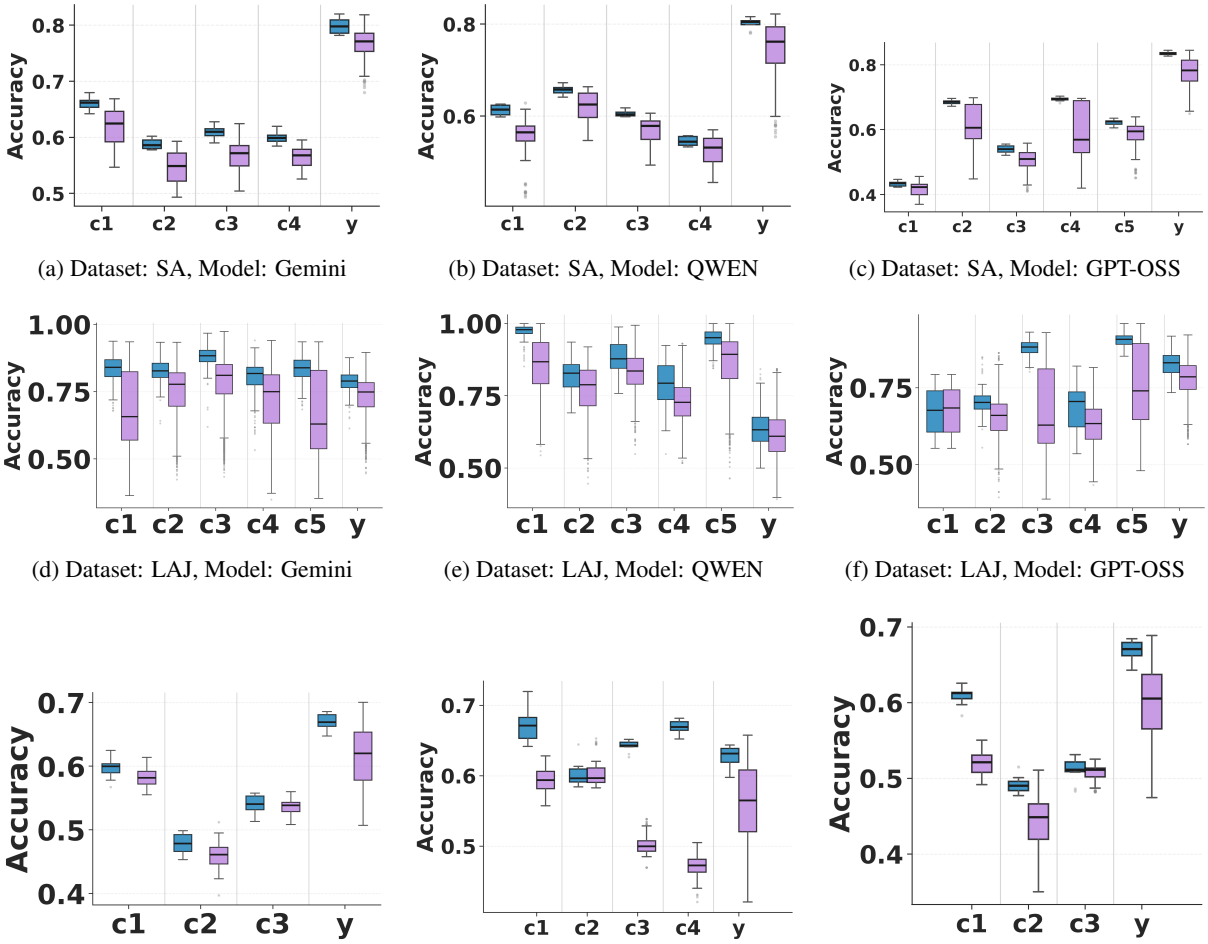
side effects: We check the non-target concept c_2. The text still says "bright orange", so $\phi(\tilde{x})[c_2] = \{\textit{tasty}\}$, matching its previous value. Thus $n_{\text{err}} = 0$. Assuming $\epsilon = 1$, the condition $n_{\text{err}} \leq \epsilon$ is satisfied.

- **Result:** The proposal is accepted, and appended to $\mathcal{D}_{\text{mcmc}}$.

Phase 4: Causal Discovery via σ-CG. After expansion, each accepted text has a concept vector and an LLM-predicted label. Since $|\mathcal{C}| = 2$ and $|\mathcal{V}| = 4$, the concept state space has $|\mathcal{V}|^{|\mathcal{C}|} = 16$ possible concept-value combinations. The expanded annotated dataset $\mathcal{D}_{\text{mcmc}}$ is passed to σ-CG. Using the background knowledge that the text node is a root and $\hat{y}$ is a sink, the algorithm outputs a directed graph $\mathcal{G} = (V, E)$ over

$$V = \{\text{text}\} \cup \{c_1, c_2\} \cup \{\hat{y}\}.$$

For visualization, we omit the text node and show the recovered dependencies among *Softness*, *Color*, and the LLM prediction.

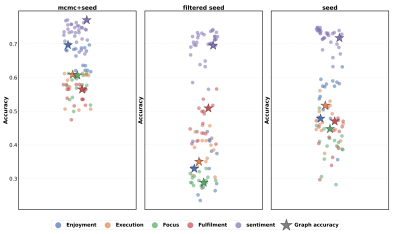


(a) Dataset: SA, Model: Gemini (b) Dataset: SA, Model: QWEN (c) Dataset: SA, Model: GPT-OSS

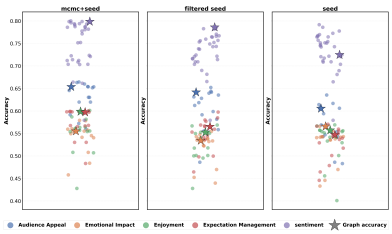
(d) Dataset: LAJ, Model: Gemini (e) Dataset: LAJ, Model: QWEN (f) Dataset: LAJ, Model: GPT-OSS

(g) Dataset: DD, Model: Gemini (h) Dataset: DD, Model: QWEN (i) Dataset: DD, Model: GPT-OSS

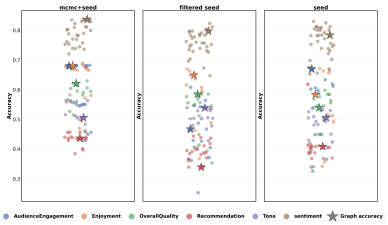
Figure 6: Box plots detailing the classification accuracy across 10 rounds of cross-validation for each model and dataset combination. To optimize space, target variables are abbreviated as c_i for intermediate latent concepts and y for the final downstream classification task. The light blue boxes represent the predictive accuracy when conditioning exclusively on the graph-specified causal parents ($Pa(v)$) derived from the representative graph topology. Conversely, the purple boxes display the accuracy distributions of all alternative predictor subsets Z that do not fully encapsulate the required parent set ($Pa(v) \not\subseteq Z$). Each data point in the underlying distribution represents a bootstrap accuracy score. Within each box plot, the central thick line denotes the median accuracy, while the bottom and top edges correspond to the 25th (Q1) and 75th (Q3) percentiles, respectively (representing the Interquartile Range, IQR, or the middle 50% of the data). The whiskers extend to the furthest data points within $1.5 \times$ IQR from the box edges. Individual grey markers denote outlier accuracy scores



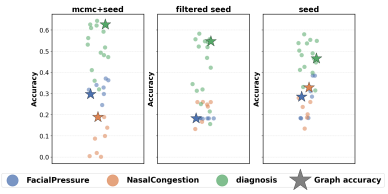
(a) Dataset: IMDB, Model: Gemini



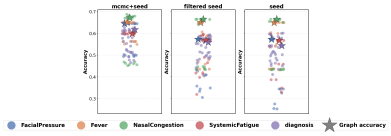
(b) Dataset: IMDB, Model: QWEN



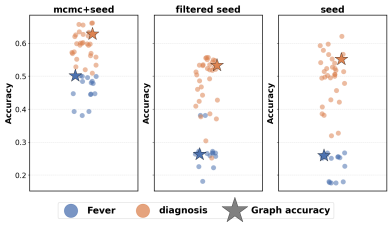
(c) Dataset: IMDB, Model: GPT-OSS



(d) Dataset: LIBERTY, Model: Gemini

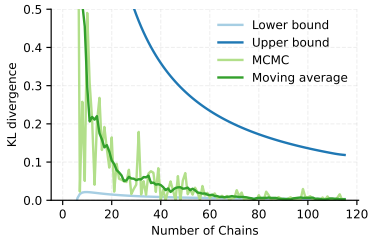


(e) Dataset: LIBERTY, Model: QWEN

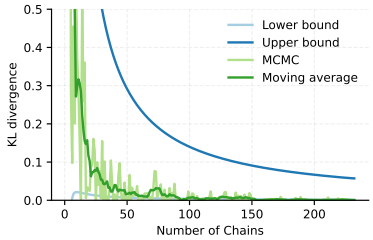


(f) Dataset: LIBERTY, Model: GPT-OSS

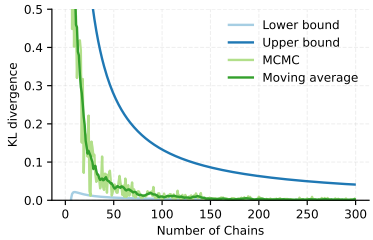
Figure 7: Concept predictability accuracy comparison across training set variations (‘mcmc+seed‘, ‘filtered seed‘, and ‘seed‘), evaluated on the MCMC-expanded test set. Colors denote specific target concepts as defined in the legend Semi-transparent, colored scatter points represent accuracies across the full combinatorial distribution of evaluated parent-child configurations for each concept. Large, colored star markers explicitly highlight the optimal accuracy specified by the learned causal graph topology; the color of each star matches the concept it evaluates .



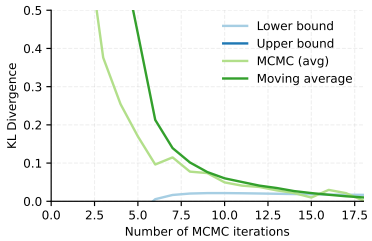
(a) Dataset: IMDB, Model: Gemini



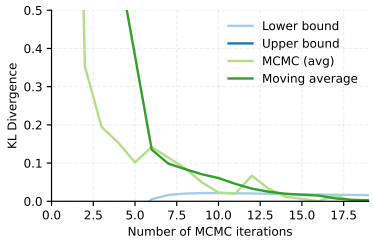
(b) Dataset: IMDB, Model: QWEN



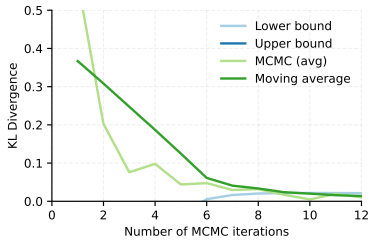
(c) Dataset: IMDB, Model: GPT-OSS



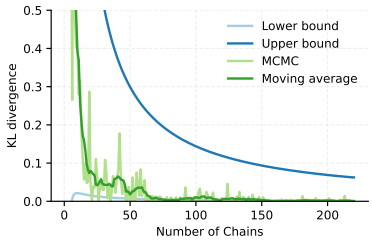
(d) Dataset: LAJ, Model: Gemini



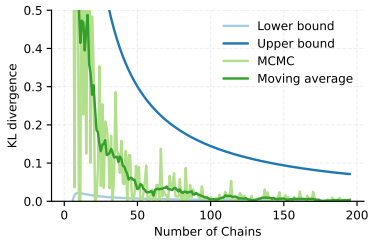
(e) Dataset: LAJ, Model: QWEN



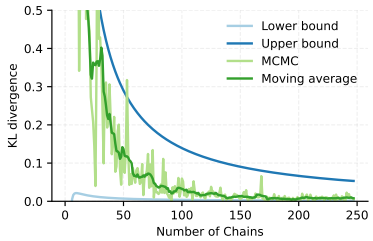
(f) Dataset: LAJ, Model: GPT-OSS



(g) Dataset: LIBERTY, Model: Gemini

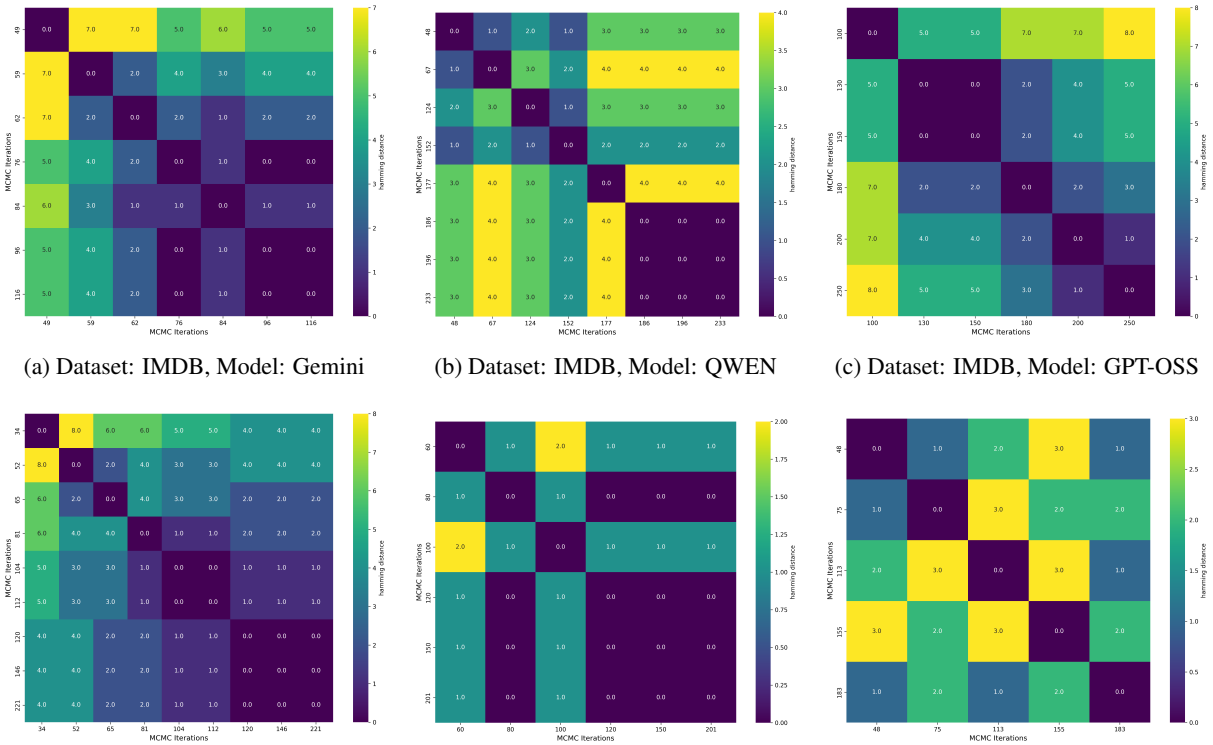


(h) Dataset: LIBERTY, Model: QWEN



(i) Dataset: LIBERTY, Model: GPT-OSS

Figure 8: The plots illustrates the convergence of our MCMC data expansion algorithm. The x-axis represents the number of MCMC iterations, where each iteration corresponds to the full expansion of a single original data instance into a new batch of samples in the IMDB and LIBERTY dataset, and the expension of on example form previous naive expension step for LAJ dataset. The y-axis denotes the Kullback-Leibler (KL) divergence between the updated cumulative probability distribution and the distribution from the previous iteration. To contextualize the empirical convergence against the natural mathematical artifact of accumulating samples, we plot two theoretical boundaries. The blue curves indicate the converged-case boundary of perfect overlap, where newly generated samples perfectly align with the previously established distribution. Conversely, the red curves signify the non converged-case boundary of orthogonal expansion, representing a scenario where each iteration introduces completely novel samples in previously unexplored regions of the concept space. The green curves represent the empirical KL divergence calculated from our generated data. Notably, the expansion and convergence processes differ across datasets. For the LIBERTY and IMDB datasets, the process is evaluated globally across all data instances, yielding a single empirical green curve per plot. In contrast, for the LAJ dataset, we construct an independent causal graph for each distinct query. Consequently, the LAJ plots display multiple green curves, each reflecting the isolated convergence of a single data item. Furthermore, because each LAJ process is restricted to the context of a single query, its latent concept distribution space is inherently smaller, which accounts for the noticeably faster convergence rate observed in these specific plots. True convergence is achieved as the empirical green curves stabilize and approach the blue boundary.



(a) Dataset: IMDB, Model: Gemini (b) Dataset: IMDB, Model: QWEN (c) Dataset: IMDB, Model: GPT-OSS

(d) Dataset: LIBERTY, Model: Gemini (e) Dataset: LIBERTY, Model: QWEN (f) Dataset: LIBERTY, Model: GPT-OSS

Figure 9: Structural consistency analysis of the learned causal models across different stages of the MCMC expansion. The figure presents six subplots, each displaying a distance matrix that compares the causal graphs generated after varying numbers of MCMC iterations. The values within the matrices represent the Hamming distance, which quantifies the exact number of differing edges between any two given graphs. This evaluation serves as a supplementary empirical validation for our MCMC convergence criterion. As the number of MCMC iterations increases, the Hamming distances between subsequently generated graphs diminish and eventually stabilize. This indicates that the topology of the causal graph achieves structural stability and ceases to change, thereby confirming that the MCMC data expansion process has successfully converged.

C Causal Graphs

This appendix presents additional causal graphs extracted by our framework across various models and datasets, complementing the results discussed in the main text.

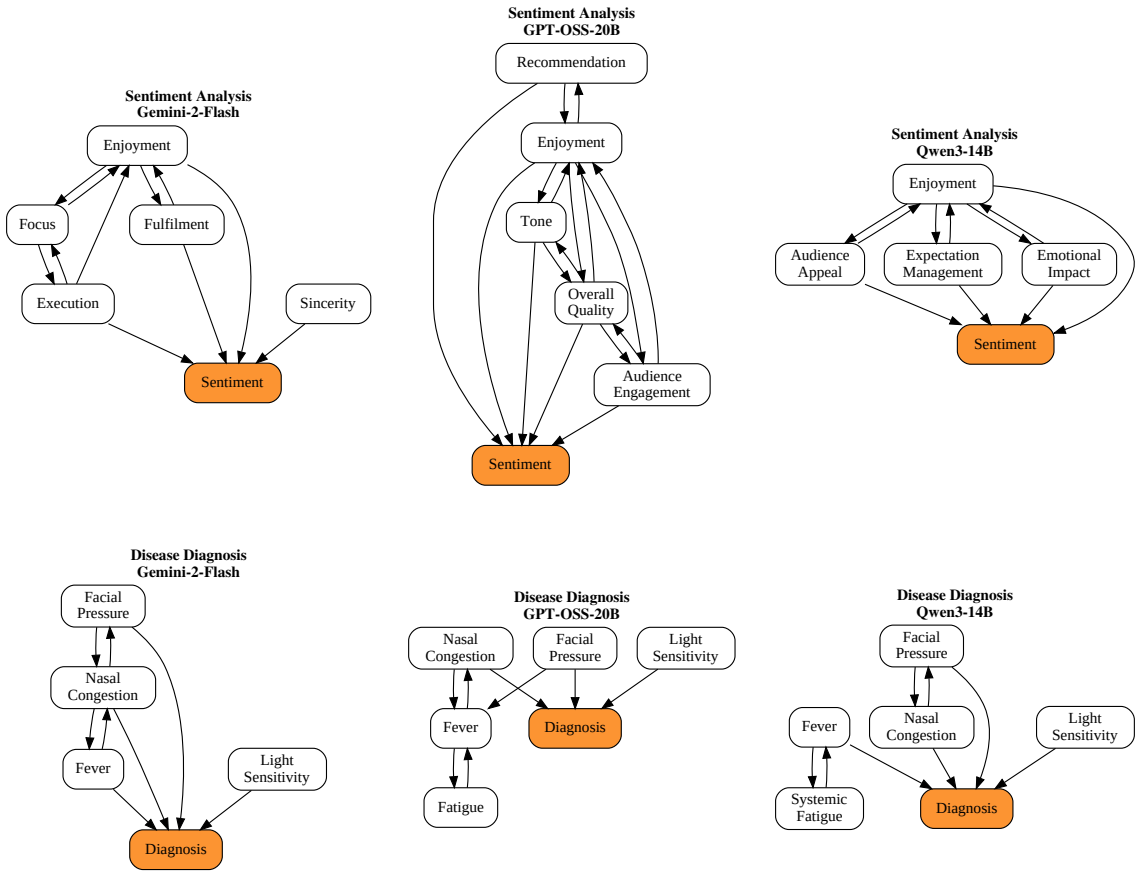


Figure 10: Extracted graphs of the SA and DD task across the different models

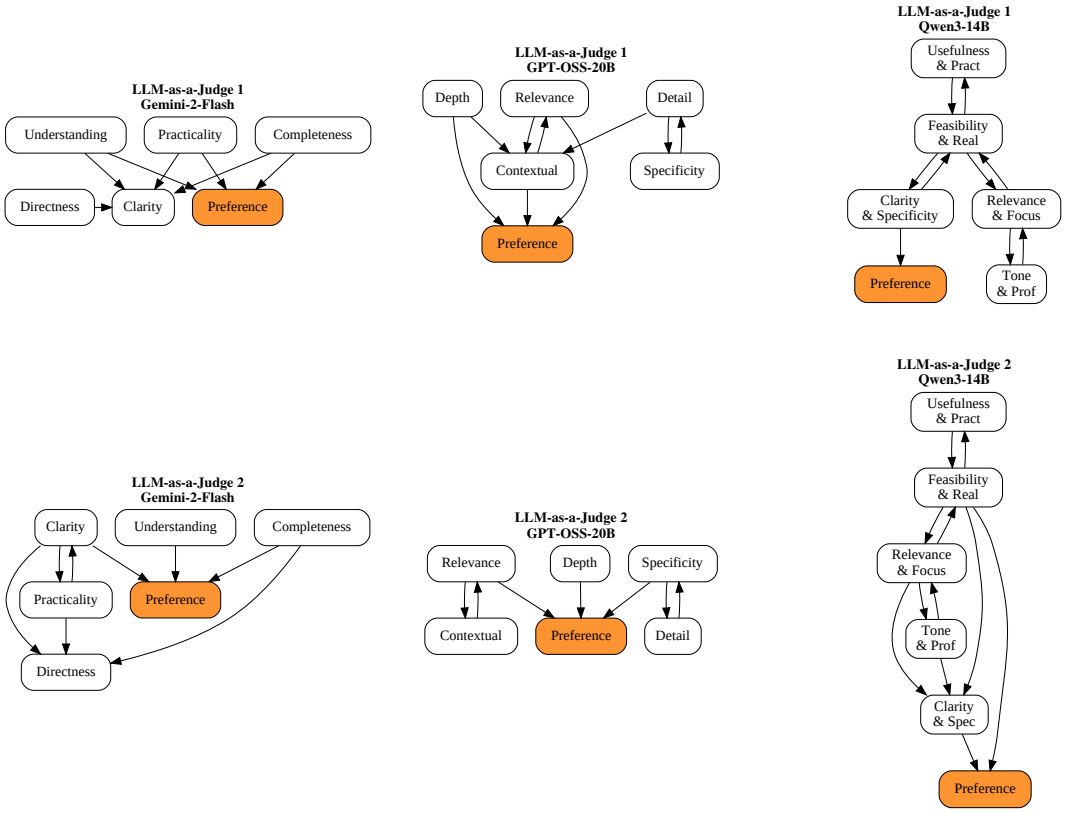


Figure 11: Representative causal graphs for the LAJ classification task across different models. Since our framework generates a unique causal graph per individual query for this dataset, we present a curated sample of these query-specific structures.

D Pseudo Algorithms

Concept Extraction The following pseudo-code outlines the Differentiative Concept Extraction phase of our framework. Because LLMs have limited context windows and output length constraints, we process data in batches and limit the active concept pool using a threshold B (set to 4 in our experiments). This limit is crucial, as too many concepts exponentially expand the text length in later steps. For each batch, the LLM first checks if the current concept set is sufficient to differentiate between the predicted classes. If not, it extracts new concepts to bridge the gap. To remove non-differentiative concepts, we prune the pool every 10 iterations. For a task with Y classes, the LLM assigns each concept one of 2^Y possible combinations. Two of these labels denoting either that a concept never appears in any class, or that it appears equally across all classes are marked as non-differentiative. The remaining $2^Y - 2$ labels represent valid discriminative combinations. A concept is kept in the pool only if its frequency of receiving a differentiative label exceeds a predefined threshold, τ. Once this extraction and filtering process concludes on the training set, the final concept set is fixed and applied to the test set.

Algorithm 1: Iterative Differentiative Concept Extraction

Input: Training set $\mathcal{D}_{train}$, Test set $\mathcal{D}_{test}$, threshold τ, max bound B
Output: Concept set $\mathcal{C}$
Initialize concept set $\mathcal{C} \leftarrow \emptyset$
foreach *iteration* $i = 1, 2, \ldots$ *and batch* $b \in \mathcal{D}_{train}$ **do**
  if $\mathcal{C}$ *insufficiently describes* b **then**
    $\mathcal{C} \leftarrow \mathcal{C} \cup$ LLM_Propose(b) (ensuring $|\mathcal{C}| \leq B$)
  end
  if $i \bmod 10 == 0$ **then**
    $\mathcal{C} \leftarrow \{c \in \mathcal{C} \mid \text{DiscriminativePower}(c, b) > \tau\}$
  end
end
$\mathcal{C} \leftarrow \{c \in \mathcal{C} \mid \text{DiscriminativePower}(c, \mathcal{D}_{test}) > \tau\}$
return Select_Top_Concepts$(\mathcal{C}, 5)$

MCMC Inspired Data Expansion The execution flow of the data expansion procedure is detailed in Algorithm 2. For each seed instance, the algorithm generates textual counterfactuals. At each step, it iterates over every concept $c \in \mathcal{C}$, uniformly samples a target class $y^* \in \mathcal{Y}$, and chooses a direction $dx \in \{\text{MORE}, \text{LESS}\}$ according to whether y^* is currently included in the subset $S \subseteq \mathcal{Y}$ mapped from the scalar concept value $\phi(x)[c]$. The target LLM proposes a counterfactual text, re-annotates it with a concept vector, and provides its annotation rationale. A proposal is accepted only if the target concept shifts in the requested direction and the number of non-target concepts that drift is at most ϵ. If a proposal is rejected, recursive refinement re-prompts the LLM with the failure feedback for up to R retries.

Algorithm 2: LLM-Guided MCMC Data Expansion with Recursive Refinement

Input : Seed dataset $\mathcal{D}$, concept set $\mathcal{C}$, class set $\mathcal{Y}$, target LLM f, annotation function ϕ, MCMC steps K, max retries R, drift threshold ϵ
Output : Expanded counterfactual dataset $\mathcal{D}_{\text{mcmc}}$

```
D_mcmc ← ∅
foreach instance x^(0) ∈ D do
    x ← x^(0)
    for k ← 1 to K do
        foreach concept c ∈ C do
            y* ~ Uniform(Y)
            ψ ← φ(x)
            S ← subset mapped from ψ[c]
            if y* ∈ S then
                dx ← LESS
            else
                dx ← MORE
            end
            x' ← f.GenerateCounterfactual(x, c, y*, dx)
            ψ', reason ← f.Annotate(x')
            S' ← subset mapped from ψ'[c]
            n_err ← Σ_{c_j ∈ C\{c}} I[ψ'[c_j] ≠ ψ[c_j]]
            aligned ← ((dx = MORE ∧ y* ∈ S') ∨ (dx = LESS ∧ y* ∉ S'))
            if aligned and n_err ≤ ε then
                x ← x'
                D_mcmc ← D_mcmc ∪ {x'}
            else
                x_rec ← RecursiveRefinement(x, ψ, x', c, y*, dx, reason, 1)
                if x_rec ≠ Null then
                    x ← x_rec
                    D_mcmc ← D_mcmc ∪ {x_rec}
                end
            end
        end
    end
end
```

Function *RecursiveRefinement*($x_{base}, \psi_{base}, x_{curr}, c, y^*, dx, reason, r$)**:**

```
if r > R then
    return Null
end
feedback ← ConstructFeedback(c, y*, dx, reason)
x_new ← f.RegenerateCounterfactual(x_curr, feedback)
ψ_new, new_reason ← f.Annotate(x_new)
S_new ← subset mapped from ψ_new[c]
n_err ← Σ_{c_j ∈ C\{c}} I[ψ_new[c_j] ≠ ψ_base[c_j]]
aligned ← ((dx = MORE ∧ y* ∈ S_new) ∨ (dx = LESS ∧ y* ∉ S_new))
if aligned and n_err ≤ ε then
    return x_new
else
    return RecursiveRefinement(x_base, ψ_base, x_new, c, y*, dx, new_reason, r + 1)
end
```

E Prompts

This section details the prompt templates utilized across the various stages of our methodology. For brevity, we present the base templates tailored for the LIBERTY dataset as representative examples, with the exception of the Preliminary Expansion stage, which was applied exclusively to the LAJ dataset. The fundamental structure of these prompts remains highly consistent across all other datasets, with only minor adaptations introduced to accommodate dataset-specific input formats. Note that these templates are static; during execution, data instances are dynamically injected in batches via an automated Python pipeline. Within these templates, placeholders such as "direction" or "DX" are dynamically populated with either "more" or "less," depending on the experimental scenario. Asterisk-bound phrases denote dynamic placeholders that are programmatically replaced by the Python pipeline during execution.

Prompt E.1: Prompt for Inference and Label Alignment stage data set : LIBERTY

Role:
You are a medical symptom classification model.

Input:
You will be given a dataset containing multiple items. Each item includes:

- `patient description`: A free-text description of a patient's symptoms.
- `item id`: An integer representing the item's original index in a larger dataset.

Task:
For each item, classify the condition described in the patient description into exactly one of the following labels:

- `MIGRAINE`
- `SINUSITIS`
- `INFLUENZA`

If the description is mixed or ambiguous, choose the most likely dominant condition.
If no dominant condition can be identified, assign `INFLUENZA` by default.

Output requirements (strict):
Your output must contain two sections, in this exact order.

Section 1 – Reasoning:
Provide a brief explanation (1–2 sentences) for each item explaining why the chosen label was selected. Do not include JSON in this section.

Section 2 – Labeled Data (JSON):
Output a single valid JSON object suitable for saving as a `.json` file.

Rules for the JSON:

- Each key must be the `item id` converted to a string.
- Each value must be an object with exactly two fields:
  - `"patient description"`: the original patient description, unchanged.
  - `"diagnosis"`: either "MIGRAINE", "SINUSITIS", or "INFLUENZA".
- Do not add comments.
- Do not add extra fields.
- Do not change the patient description text.
- Do not output multiple JSON objects.

After the JSON object is closed, print exactly:
END JSON

Example output:

{

"101": {
"patient description": "Severe throbbing headache on the right side with nausea and sensitivity to light.",
"diagnosis": "MIGRAINE"
},
"102": {
"patient description": "Facial pressure, congestion, and thick nasal discharge for several days.",
"diagnosis": "SINUSITIS"
},
"103": {
"patient description": "Sudden fever, muscle aches, cough, and extreme fatigue.",
"diagnosis": "INFLUENZA"
}
}

END JSON

Dataset:

Prompt E.2: Prompt for first iteration of concept Differentiative Concept Extraction stage dataset : LIBERTY

Role:
You are an analyst extracting dataset-level features for medical condition differentiation.

Input:
You will receive a dataset of patient cases. Each item contains:

- `patient_description`: a free-text description of a patient's symptoms
- `disease`: one of the labels "MIGRAINE", "SINUSITIS", or "INFLUENZA" (these labels were assigned previously)

Task: Symptom Extraction
Across the entire dataset, extract exactly "B#" DISTINCT SYMPTOMS that help distinguish between the different diseases.

Definition of a Symptom:
A Symptom is a concrete, clinically meaningful manifestation explicitly described in patient text (e.g., headache, fever, nausea, nasal congestion). Symptoms should be observable or reportable experiences, not abstract properties or disease names.

Requirements:

- **Differentiation:** Each symptom must clearly distinguish between at least two of the diseases. For every symptom, it must be true that the symptom is typically more frequent, more prominent, or more characteristic in one disease than in at least one of the other diseases.
- **Classification usefulness:** The symptoms must be phrased so they can be identified in new patient descriptions whose disease label is unknown.
- **Count:** Extract exactly "B#" symptoms (no more, no less). Do NOT use the number of symptoms from any toy/example—use only "B#".
- **Naming:** Give each symptom a short, precise name, preferably a single word or short clinical phrase (e.g., Headache, Fever, Nausea, Congestion).

- **Avoid trivial or circular symptoms:** Do not list disease names or symptoms that simply restate the diagnosis (e.g., "Migraine pain"). Focus on concrete symptoms that generalize across patients.

Output format (strict):
Your output must contain two sections, in this exact order.

Section 1 – Reasoning:
Briefly explain (1–3 sentences per symptom) why each symptom helps distinguish between the diseases based on patterns observed across the dataset.

Section 2 – Symptoms JSON:
Output a single valid JSON object suitable for saving as a `.json` file. Each key is a symptom name. Each value is a concise description explaining how the symptom differs across MIGRAINE, SINUSITIS, and/or INFLUENZA.

After the JSON object is closed, print exactly:
END JSON

Toy Example (for illustration only):

Toy Input Dataset (number of symptoms to extract = 2):

Item 1:
`patient_description`: "Severe headache with nausea and sensitivity to light."
`disease`: MIGRAINE

Item 2:
`patient_description`: "Facial pressure and nasal congestion lasting several days."
`disease`: SINUSITIS

Item 3:
`patient_description`: "High fever, muscle aches, and extreme fatigue."
`disease`: INFLUENZA

Toy Output (number of symptoms to extract = 2):

Reasoning:
Headache: Head pain is described across diseases but differs in accompanying features and prominence, making it useful for differentiation.
Fever: Fever appears primarily in systemic illness descriptions and is absent or rare in others, providing a strong distinguishing signal.

{
"Headache": "Head pain is described across diseases but varies in context and co-occurring symptoms.",
"Fever": "Fever appears mainly in systemic illness descriptions and is uncommon in others."
}
END JSON

Dataset:

Prompt E.3: Prompt for Differentiative Concept Extraction stage dataset : LIBERTY

Role:
You are an analyst extracting dataset-level features for medical condition differentiation.

Input:

You will receive a dataset of patient cases. Each item contains:

- `patient_description`: a free-text description of a patient's symptoms
- `disease`: one of the labels "MIGRAINE", "SINUSITIS", or "INFLUENZA" (these labels were assigned previously)

You are also provided with a set of existing SYMPTOMS and their descriptions.

Task: Symptom Refinement and Optional Augmentation
Your primary task is to critically review the provided dataset and determine whether the existing symptoms are sufficiently comprehensive to effectively and completely distinguish between MIGRAINE, SINUSITIS, and INFLUENZA across the entire dataset.
If — and ONLY IF — the existing symptoms are not sufficient, you may augment them by extracting additional symptoms from the dataset.

Important:

- You are NOT required to add new symptoms.
- You may add FEWER than "B#" symptoms.
- You may add ZERO new symptoms if the existing set is already sufficient.
- Adding unnecessary or redundant symptoms is worse than adding none.
- Only extract additional symptoms if they provide a clear and meaningful improvement in distinguishing between the diseases.

Definition of a Symptom:
A Symptom is a concrete, clinically meaningful manifestation explicitly described in patient text (e.g., headache, fever, nausea, nasal congestion). Symptoms must be grounded in explicit textual evidence and must generalize across multiple patients.

Requirements:

- **Differentiation:** Each newly added symptom must clearly help distinguish between at least two of the diseases. For every added symptom, it must be true that the symptom is more frequent, more prominent, or more characteristic in one disease than in at least one of the other diseases.
- **Necessity:** Only add a symptom if its absence would leave a meaningful gap in disease differentiation. Do NOT add symptoms that are redundant with existing ones or that add minimal new information.
- **Classification usefulness:** All symptoms (existing and newly added) must be usable for classifying new, unseen patient descriptions whose disease label is unknown.
- **Count:** You may add up to "B#" additional symptoms. You may add fewer than "B#" symptoms, or none at all. Do NOT force the number of added symptoms to reach "B#".
- **Naming:** Give each symptom a short, precise, clinically meaningful name. Avoid vague or abstract names. Do not include disease names in symptom names.
- **Avoid trivial or circular symptoms:** Do not restate disease labels. Do not add overly specific or one-off symptoms that do not generalize across patients.

Output format (strict):
Your output must contain two sections, in this exact order.

Section 1 – Reasoning:
Briefly explain (1–3 sentences per symptom) whether the existing symptoms are sufficient. If new symptoms are added, explain why each added symptom is necessary and what differentiating gap it fills. If no new symptoms are added, explicitly state that the existing symptoms are sufficient.

Section 2 – Symptoms JSON:
Output a single valid JSON object suitable for saving as a `.json` file.
The JSON must include:

- All existing symptoms (unchanged)
- Any newly added symptoms (if applicable)

Each key is a symptom name. Each value is a concise description explaining how the symptom differs across MIGRAINE, SINUSITIS, and/or INFLUENZA.
Do not add comments. Do not add extra fields.

After the JSON object is closed, print exactly:
END JSON

Toy Example (for illustration only):

Existing Symptoms (provided to the model):
Localization: Describes where symptoms are primarily experienced in the body.

Toy Input Dataset (maximum number of symptoms to add: 2):

Item 1:
`patient_description`: "Throbbing headache on one side with nausea and light sensitivity."
`disease`: MIGRAINE

Item 2:
`patient_description`: "Facial pressure, congestion, and pain around the cheeks for several days."
`disease`: SINUSITIS

Item 3:
`patient_description`: "Sudden fever, muscle aches, cough, and extreme fatigue."
`disease`: INFLUENZA

Explanation of insufficiency (implicit in reasoning):
The existing symptom Localization captures where symptoms occur but does not capture differences in whole-body involvement or temporal onset.

Toy Output (number of symptoms added: 2):

Reasoning:
Localization: Symptoms are concentrated in different body regions across diseases, but this alone does not capture differences in systemic involvement.
SystemicSymptoms: Some diseases involve widespread bodily effects such as fatigue and aches, which are not captured by localization alone.
OnsetSpeed: Descriptions differ in whether symptoms begin suddenly or develop gradually, providing an additional distinguishing signal.

{
"Localization": "Symptoms differ in where discomfort is concentrated across diseases.",
"SystemicSymptoms": "Some diseases involve widespread bodily symptoms, while others are

primarily localized.",
"OnsetSpeed": "Symptoms differ in whether they appear suddenly or develop over time."
}
END JSON

Dataset:
Previous Symptoms:

Prompt E.4: Prompt for Concept Annotation stage dataset : LIBERTY

Role:
You are a symptom-level medical classification annotator.

Input:
You will receive:

- A dataset of patient cases. Each item contains:
  - `item_id`: an integer representing the item's original index
  - `patient_description`: a free-text description of a patient's symptoms
- A list of SYMPTOMS. Each symptom includes:
  - `symptom_name`: a short, precise name
  - `symptom_description`: a description explaining what the symptom captures as a clinical manifestation

Task:
For each dataset item, assign a label to each SYMPTOM based on:

- Whether the symptom appears in the `patient_description` at all.
- If it appears, which disease or diseases the symptom is most consistent with, given the context of the description.

Disease set:

- MIGRAINE
- SINUSITIS
- INFLUENZA

Label definitions (important – read carefully):

- **0** → The symptom does NOT appear in the text at all. There is no explicit mention or clear reference to this symptom in the `patient_description`.
- **1** → The symptom appears in the text and fits MIGRAINE only.
- **2** → The symptom appears in the text and fits SINUSITIS only.
- **3** → The symptom appears in the text and fits INFLUENZA only.
- **4** → The symptom appears in the text and fits BOTH MIGRAINE and SINUSITIS.
- **5** → The symptom appears in the text and fits BOTH MIGRAINE and INFLUENZA.

- **6** → The symptom appears in the text and fits BOTH SINUSITIS and INFLUENZA.
- **7** → The symptom appears in the text, BUT it does NOT meaningfully differentiate between MIGRAINE, SINUSITIS, and INFLUENZA. Use this label ONLY when the symptom is explicitly mentioned and is equally consistent with all three diseases given the description.

Important rules:

- Assign exactly one label (0–7) for every symptom in every item.
- Label 0 must be used ONLY when the symptom is completely absent from the text. If the symptom appears even vaguely, label 0 is NOT allowed.
- Label 7 must be used ONLY when the symptom appears in the text AND cannot be used to distinguish between the three diseases.
- Use the `symptom_description` together with the `patient_description` to decide whether the symptom is present, and whether it differentiates between diseases or not.
- A symptom may appear regardless of the final diagnosed disease.
- Do NOT infer the presence of a symptom unless it is explicitly supported by the text.
- Disease compatibility should be judged based on how the symptom is described (e.g., location, quality, duration, context), not on the dataset label.
- If a symptom clearly fits multiple diseases, select the label corresponding to the full set of diseases it fits. If it fits all three equally, select label 7.

Output format (strict):
Your output must contain two sections, in this exact order.

Section 1 – Reasoning:
For each dataset item, briefly explain the reasoning behind the assigned labels. The reasoning should reference:

- evidence from the `patient_description`
- why each symptom was considered present or absent
- why the symptom does or does not differentiate between diseases

Keep explanations concise (1–2 sentences per symptom per item). Do NOT include JSON in this section.

Section 2 – Item Symptom Output (JSON):
Output a single valid JSON object suitable for saving as a `.json` file.

JSON rules:

- Each key is the `item_id` converted to a string.
- Each value is an object whose keys are the `symptom_name` values and whose values are strings in {"0", "1", "2", "3", "4", "5", "6", "7"}.
- All symptoms must appear for every item.
- Do not add extra fields or comments.

After the JSON object is closed, print exactly:
END JSON

Dataset:

Prompt E.5: Prompt for Preliminary Expansion stage dataset : LAJ

Role:
You are a preference optimization and data alignment expert.

Core Task:
Your goal is to make the concept: `"*Target Concept*"` associate with the `"*direction*"` Response.

The description of "*Target Concept*": "`*Concept description*`"

For EACH item, strictly follow this:
Check `"*Target Concept*"` Current Label. Based ONLY on this label:

- **IF Current Label IS `"*wanted label*"`:**
  - DO NOT CHANGE Chosen Response or Rejected Response.
  - `ChangeFlag` = False.
- **IF Current Label IS NOT `"*wanted label*"`:**
  - You MUST MODIFY both Chosen Response and/or Rejected Response.
  - **Modification Goal:** Make Chosen Response `"*directionChosen*"` aligned with `*Target Concept*`, and make Rejected Response `"*directionRejected*"` aligned with `*Target Concept*`.
  - **Guidelines for Mandatory Modifications:**
    - *Plausibility:* Responses must remain logical and coherent.
    - *Subtlety (if possible):* Aim for minimal effective changes.
  - `ChangeFlag` = True.
  - **Reasoning (MANDATORY if modified):** Explain the exact text changes and how they make `"*Target Concept*"` fit Chosen Response and Rejected Response.

Input:
Dataset items, each with `Original Index`, `User Query`, `Chosen Response`, `Rejected Response`, and `Concept Labels` (telling you the Current Label for `*Target Concept*`).

Output Format:

1. JSON Output:
A single valid JSON object. Item keys are "Item" + Original Index (e.g., "Item98").
JSON Structure:

```
{
  "Item_Original_Index_Here" : {
    "User Query": "Original user query",
    "Original Index": "Original Index from input",
    "Chosen Response": "Final Chosen Response",
    "Rejected Response": "Final Rejected Response",
    "ChangeFlag": boolean
  }
}
END JSON
```

2. Reasoning Output (After JSON, one block per item):

```
Reasoning for Item [Original Index]:
```

```
1. Original Index: [Original Index]
2. Target Concept: [*Target Concept* Name]
3. Current Label for Target Concept: [1, 0, 2, or 3]
4. Target Label: "*wanted label*"
5. Critical Check Outcome:
   ["Labels are identical (Current Label is
   "*wanted label*")" OR "Labels are NOT identical
   (Current Label is [actual_label], not
   "*wanted label*")"]
6. Scenario Applied:
   ["Scenario A applied (because Current Label is
   "*wanted label*")" OR "Scenario B applied
   (because Current Label is [actual_label],
   not "*wanted label*")"]
7. Final Action & ChangeFlag Value Details:
   - If Scenario A: "No modifications made as
     required by Scenario A. ChangeFlag is False."
   - If Scenario B: "Modifications were mandatory
     under Scenario B. [Describe changes to Chosen
     and Rejected responses.]"
   ChangeFlag is True.
8. If you were given the reason for current labeling
   by the LLM, add how you included this reason in
   the modified responses.
```

EXAMPLE (Keep this output structure and reasoning detail):

Let's assume the Target Concept is "Conciseness".

Input Item 1:
`Original Index:` 98
`User Query:` "What is the capital of France?"
`Chosen Response:` "Ah, you're asking about the capital of France! Paris Well, let me tell you, the truly magnificent, beautiful, and significantly historic capital city of the glorious French Republic, which is situated in the lovely north-central part of that European country, is none other than Paris, a global center for art, fashion, gastronomy and culture."
`Rejected Response:` "Paris."
`Concept Labels:` {"Conciseness": 3}

Input Item 2:
`Original Index:` 99
`User Query:` "Explain photosynthesis briefly."
`Chosen Response:` "Plants make food from sun."
`Rejected Response:` "Photosynthesis is the process plants use to convert light energy into chemical energy (food), using water, carbon dioxide, and sunlight, releasing oxygen as a byproduct. It's vital for life on Earth."
`Concept Labels:` {"Conciseness": 1}

Expected JSON Output:

```
{
  "Item98": {
    "User Query": "What is the capital of France?",
```

```
    "Original Index": "98",
    "Chosen Response": "The beautiful and historic capital
     city of the French Republic, located in the
     north-central part of the country, is Paris.",
    "Rejected Response": "The beautiful and historic Paris.",
    "ChangeFlag": true
  },
  "Item99": {
    "User Query": "Explain photosynthesis briefly.",
    "Original Index": "99",
    "Chosen Response": "Plants make food from sun.",
    "Rejected Response": "Photosynthesis is the process...",
    "ChangeFlag": false
  }
}
END JSON
```

Expected Reasoning Output:

```
Reasoning for Item 98:
1. Original Index: 98
2. Target Concept: Conciseness
3. Current Label for Target Concept: 3
4. Target Label: 1
5. Critical Check Outcome: Labels are NOT identical
   (Current Label is 3, not 1).
6. Scenario Applied: Scenario B applied (because
   Current Label is 3, not 1).
7. Final Action & ChangeFlag Value Details:
   Modifications were mandatory under Scenario B.
   The original Rejected Response ("Paris.") was made
   significantly less concise by adding verbiage.
   The original Chosen Response (a long sentence)
   was made more concise. These changes aim to ensure
   "Conciseness" now better represents Chosen.
   ChangeFlag is True.

Reasoning for Item 99:
1. Original Index: 99
2. Target Concept: Conciseness
3. Current Label for Target Concept: 1
4. Target Label: 1
5. Critical Check Outcome: Labels are identical
   (Current Label is 1).
6. Scenario Applied: Scenario A applied (because
   Current Label is 1).
7. Final Action & ChangeFlag Value Details:
   No modifications made as required by Scenario A.
   ChangeFlag is False.
```

Dataset:

Prompt E.6: Prompt for MCMC Inspired Expansion stage dataset : LIBERTY

Role:
You are a medical data augmentation and modification expert.

Core Task:
Your goal is to modify a patient description so that a symptom: `*SYMPTOM*` becomes `*DX*` aligned with a `*DISEASE*`.

Target Symptom description:
`*SYMPTOM* : *SYMPTOM_DESCRIPTION*`

Symptom labels and their meanings:

- **0** → The symptom does NOT appear in the text at all. There is no explicit mention or clear reference to this symptom in the `patient_description`.
- **1** → The symptom appears in the text and fits MIGRAINE only.
- **2** → The symptom appears in the text and fits SINUSITIS only.
- **3** → The symptom appears in the text and fits INFLUENZA only.
- **4** → The symptom appears in the text and fits BOTH MIGRAINE and SINUSITIS.
- **5** → The symptom appears in the text and fits BOTH MIGRAINE and INFLUENZA.
- **6** → The symptom appears in the text and fits BOTH SINUSITIS and INFLUENZA.
- **7** → The symptom appears in the text, BUT it does NOT meaningfully differentiate between MIGRAINE, SINUSITIS, and INFLUENZA. Use this label ONLY when the symptom is explicitly mentioned and is equally consistent with all three diseases.

For EACH item, strictly follow this: You MUST MODIFY the `patient_description` if alignment is requested.

Modification Goal:
Make the `patient_description` `*DX*` aligned with the Target Symptom for the `*DISEASE*` disease.

Critical Rule – NO DISEASE MENTION (MANDATORY):

- You MUST NOT mention, suggest, imply, or hint at the disease name or diagnosis in the `patient_description`.
- Do NOT use disease names (e.g., "migraine", "sinusitis", "flu").
- Do NOT use phrases that directly suggest a diagnosis (e.g., "consistent with migraine", "suggestive of flu").
- Alignment must be achieved ONLY through symptom characteristics (e.g., location, quality, duration, triggers, severity).

Guidelines for Mandatory Modifications:

- **Plausibility:** The modified `patient_description` must remain medically realistic and coherent.
- **Subtlety (if possible):** Aim for minimal but effective textual changes.

- **Symptom isolation:** Make a strong effort NOT to change the presence or interpretation of other symptoms.
- Do NOT introduce new unrelated symptoms unless absolutely necessary.
- `ChangeFlag` must be set to True if any modification is made.

Reasoning (MANDATORY if modified):
You must explain:

- What exact textual changes were made
- How those changes increase or decrease alignment of the Target Symptom with the `*DISEASE*` disease
- Why the modification respects the label definitions above
- Confirm explicitly that no disease name or diagnosis was mentioned in the `patient_description`

Input:
Dataset items, each containing: `Original Index`, `patient_description`, and the `Current label for the target Symptom (0-7)`.

Output Format:

JSON Output:
Output a single valid JSON object. Item keys are "Item" + Original Index (e.g., "Item42").
JSON structure:

```
{
  "Item_Original_Index_Here": {
    "Original_index": "Original Index from input",
    "Chosen Disease": "MIGRAINE | SINUSITIS | INFLUENZA",
    "Target Symptom": "Target Symptom name",
    "Original Patient Description": "Original description",
    "Modified Patient Description": "Final description",
    "ChangeFlag": true
  }
}
END JSON
```

Reasoning Output (after JSON, one block per item):

```
Reasoning for Item [Original Index]:
Original Index: [Original Index]
Target Symptom: [Target Symptom]
Current Label: [0--7]
Chosen Disease: [Disease]
Desired Direction: [MORE aligned / LESS aligned]
```

Final Action:
Describe the precise textual edits and explain how they change the alignment of the Target Symptom with the chosen disease without mentioning or implying the disease, while keeping other symptoms stable.

EXAMPLE (fully worked example):

Input values for this example:

- **Target Symptom:** Headache
- **Symptom description:** Describes the presence and characteristics of head pain.
- **Chosen disease:** MIGRAINE
- **Target alignment:** MORE aligned
- **Current Label:** 0

Original patient_description:
"Patient reports mild fatigue and nausea over the past day, but no other significant complaints."

Explanation of the setup:
The current label is 0, meaning headache does not appear at all in the original text. Target alignment = MORE aligned and DISEASE = MIGRAINE. The description must be modified to introduce a headache pattern typical of migraine without mentioning or suggesting migraine. Other symptoms (fatigue, nausea) must remain unchanged.

Expected JSON Output:

```
{
  "42": {
    "Original_index": "42",
    "Chosen Disease": "MIGRAINE",
    "Target Symptom": "Headache",
    "Original Patient Description": "Patient reports mild
     fatigue and nausea over the past day, but no other
     significant complaints.",
    "Modified Patient Description": "Patient reports mild
     fatigue and nausea over the past day, along with
     a throbbing headache on one side of the head that
     worsens with light exposure.",
    "ChangeFlag": true
  }
}
END JSON
```

Expected Reasoning Output:

```
Reasoning for Item 42:
Original Index: 42
Target Symptom: Headache
Current Label: 0
Chosen Disease: MIGRAINE
Desired Direction: MORE aligned

Final Action:
A unilateral, throbbing headache with light sensitivity
was added to the patient_description, which increases
alignment of the headache symptom with migraine-related
```

```
patterns. No disease names or diagnostic language were
introduced, and existing symptoms were preserved
unchanged, fully respecting the label definitions.
```

Dataset:

F Artifact Licenses and Usage

In compliance with reproducibility and ethical guidelines regarding artifact licenses, we detail the licensing, terms of use, and intended usage of the models and datasets utilized in this study. All computational artifacts were employed strictly for academic research purposes, consistent with their respective licenses.

Models

- **Gemini 2 Flash** (Team et al., 2025): Accessed via the official API. Usage complies with Google's Terms of Service and generative AI usage guidelines for research and evaluation purposes.
- **Qwen3-14B** (Yang et al., 2025): The model weights and inference code are utilized under the Apache 2.0 License, which explicitly permits academic research applications.
- **gpt-OSS-20b** (OpenAI, 2025): Utilized under its respective open-source license Apache 2.0, adhering to all research usage constraints.

Datasets

- **LIBERTY (Disease Diagnosis)** (Toker et al., 2026): A synthetic medical corpus. The dataset is publicly released by the authors for research purposes; while no explicit license is designated in the original publication, it is utilized here strictly for non-commercial academic research.
- **IMDB (Sentiment Analysis)** (Maas et al., 2011): The dataset is publicly available for research purposes and is utilized in accordance with its standard academic distribution terms.
- **LAJ (LLM-as-a-Judge)** (Calderon et al., 2025): Derived from Reddit data, the dataset is publicly released by the authors for research purposes. While no explicit open-source license is provided in the original publication, our usage strictly complies with the Reddit API terms of service and is restricted entirely to non-commercial academic research.

Code and Framework The code for our concept extraction, MCMC-based counterfactual generation, and causal graph construction is open-sourced and released under the MIT License to facilitate unrestricted use and reproducibility by the research community.

G Computational Resources and Compute Cost

Hardware Infrastructure Experiments evaluating the open-weight models (Qwen3-14B and gpt-OSS-20b) were conducted on an institutional remote compute cluster equipped with NVIDIA H200 and RTX6K GPUs. To optimize inference throughput during the intensive data expansion phase, these models were served utilizing the vLLM (Kwon et al., 2023) framework. Gemini-2-Flash was accessed externally via its official API.

Cost Estimation The primary computational bottleneck in our proposed framework is the MCMC-inspired counterfactual data expansion, which necessitates repeated, iterative LLM sampling.

- **API Costs (Gemini):** Executing the complete concept extraction and MCMC expansion pipeline via the Gemini API incurred a total cost of approximately $540 USD.
- **Local GPU Compute (Qwen3 & gpt-OSS):** Generating the counterfactuals for the open-weight models on our institutional cluster incurred negligible marginal compute costs (under $5 USD in equivalent compute time).

Importantly, once the counterfactual datasets are generated, the downstream causal discovery phase training the multinomial logistic regression models to extract the graph topologies is computationally lightweight and can be executed on standard CPU infrastructure within minutes.

H AI Assistance Statement

In accordance with the ACL guidelines regarding the use of AI, we acknowledge the use of AI assistants during the preparation of this work. Specifically, GitHub Copilot was utilized to assist in writing data visualization scripts and refactoring portions of the codebase for asynchronous execution. Additionally, AI language models (Gemini, ChatGPT, and Claude) were used as writing assistants to refine phrasing and improve readability. The authors take full responsibility for all content, ideas, and code presented in this paper.