Towards Probabilistic Question Answering Over Tabular Data

Chen Shen
Megagon Labs
chen_s@megagon.ai

Estevam Hruschka
Megagon Labs
estevam@megagon.ai

Abstract

Progress in question answering (QA) over tabular data has enabled reliable factual retrieval from relational tables. However, many real-world questions are probabilistic, requiring reasoning under uncertainty and latent conditional dependencies that are not directly stored in individual cells. We introduce LUCARIO, a large-scale benchmark for probabilistic QA over real-world tabular datasets, covering diverse domains, dependency patterns, and natural language variations. We propose Auto-BN, a fully automatic neuro-symbolic framework that induces a Bayesian Network from raw tables, translates natural-language questions into formal probabilistic queries, and performs exact inference. Experiments across multiple LLM backbones show that Auto-BN consistently outperforms NL2SQL, retrieval-based, and premise-based baselines, while maintaining a near-zero error rate. LUCARIO fills a relevant gap and provides a practical and scalable testbed for advancing probabilistic reasoning over structured data.

1 Introduction

Tabular data is frequently used to support decision-making across enterprise, scientific, and web applications, from CRM systems and healthcare records to logistics and financial analysis. With the rapid advancement of large language models (LLMs), question answering (QA) over structured data has emerged as a critical research frontier. Recent progress in natural language to SQL (NL2SQL or Text2SQL) systems has enabled accurate retrieval of factual information from relational tables (Talaei et al., 2024; Shen et al., 2024; Li et al., 2024). Factual queries such as "*What is the customer's signup date?*" can now be reliably answered by mapping natural language to SQL queries that extract precise values directly from the database.

Yet, many real-world information needs extend far beyond factual retrieval. Questions such as "*What is the likelihood that a customer will churn given their recent activity?*" or "*How probable is a delayed shipment if the warehouse is in Region A?*" require reasoning under uncertainty. The correct answers to these probabilistic queries involve incomplete evidence, latent and conditional dependencies that are usually not directly encoded in a single table cell, neither fully available in LLMs' parametric knowledge. Thus, current SQL-based and LLM-only approaches fall short in this setting, often hallucinating answers (Huang et al., 2025) or generalizing poorly when tasked with probabilistic reasoning (Chakraborty et al., 2025). The answers to these questions are estimated from the records of a specific table, so the same churn question can have a different answer for a different CRM table.

Prior work (Jin et al., 2023; Schrader et al., 2024; Nafar et al., 2025) has explored probabilistic graphical models and causal inference frameworks for addressing uncertainty and probabilistic reasoning jointly with LLMs. These approaches are constrained by scalability and practicality: they typically operate only on small or synthetic datasets and rely on expert-curated structures for the underlying models. In contrast, real-world tabular data may span hundreds of attributes and tens of thousands of rows, rendering manual construction of probabilistic models infeasible. Compounding this challenge is the absence of benchmarks for probabilistic QA over large-scale tabular data. Existing benchmarks overwhelmingly target factual QA, leaving no reliable way to evaluate, compare, or systematically advance systems designed for probabilistic reasoning.

In this paper, we introduce LUCARIO, a large-scale benchmark and a framework for probabilistic QA over tabular data. Our main contributions are:

1. A new benchmark with 10 real-world datasets, designed to capture diverse probabilistic reasoning patterns beyond factual retrieval and realistic scale.
2. Auto-BN, a fully automatic neuro-symbolic

Proceedings of the 15th International Joint Conference on Natural Language Processing and the 5th Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics, pages 1–23
November 6-10, 2026 ©2026 Association for Computational Linguistics

framework that (i) induces observational Bayesian Networks (BNs) directly from large relational tables, (ii) robustly translates natural language into formal probabilistic queries, and (iii) performs exact probabilistic inference equipped with a multi-level semantic relaxation.

3. Extensive empirical analysis and evaluation across diverse baselines and modeling paradigms, demonstrating consistent improvements and providing insights into the challenges and opportunities of probabilistic QA over large-scale tabular data.

Our results show that the proposed hybrid approach not only improves accuracy on complex probabilistic questions but also enhances interpretability and robustness. Beyond the proposed system, LUCARIO fills a key gap in existing resources by providing a realistic and scalable testbed for evaluating probabilistic question answering over tabular data, supporting systematic research on methods that must reason under uncertainty.

2 Related Work

Question Answering over Tabular Data: Question answering (QA) over tabular data has become an increasingly active research area, with models such as TAPAS (Herzig et al., 2020), TURL (Deng et al., 2022), and Chase-SQL (Pourreza et al., 2024) demonstrating strong performance on factual queries by translating natural language into structured queries like SQL. These systems excel when answers can be directly retrieved from table cells, e.g., *"What is the customer's signup date?"*. Nonetheless, they struggle with queries requiring reasoning beyond direct retrieval—particularly those involving uncertainty or probabilistic dependencies.

Recent systems have incorporated more complex reasoning pipelines. TQAgent (Zhao et al., 2025) integrates knowledge graphs and tree-structured reasoning paths to handle multi-step queries across diverse table schemas. TableBench (Wu et al., 2025) spans fact checking, numerical reasoning, data analysis, and visualization, and TReB (Li et al., 2026) defines 26 table-reasoning sub-tasks; neither targets conditional-probability questions answered from a joint distribution induced from the table. While such efforts extend the scope of tabular QA, they remain ill-suited for probabilistic reasoning tasks that demand modeling latent dependencies and conditional relationships not explicitly represented in the data.

Benchmarks for Probabilistic Dependency Reasoning: Evaluating how language models reason about uncertainty and dependencies among variables has motivated the development of several recent benchmarks.

Some benchmarks explicitly focus on causal reasoning. **CausalQA** (Bondarenko et al., 2022) and **QRData** (Liu et al., 2024) evaluate models' ability to identify causal structures and reason about interventions and counterfactuals. These datasets target reasoning tasks grounded in causal graphical models and structural assumptions about data-generating processes.

Other benchmarks emphasize probabilistic reasoning more broadly, often relying on Bayesian networks or textual descriptions of probabilistic relationships. **BLInD** (Nafar et al., 2025) provides natural language queries grounded in predefined Bayesian Networks (BNs), requiring models to reason over structured uncertainty and produce numeric probability estimates. **CLADDER** (Jin et al., 2023) evaluates reasoning across different levels of dependency queries, including associational and causal questions derived from symbolic graphs. **QUITE** (Schrader et al., 2024) tests Bayesian reasoning using textual descriptions of probabilistic relationships, assessing models' ability to quantify uncertainty from structured descriptions.

While impactful, these benchmarks share notable limitations. They typically rely on small, predefined Bayesian Networks (e.g., QUITE networks average only 6 nodes and 3 states per node, yielding fewer than 50 unique one-hop reasoning paths on average). This restricts both the diversity of reasoning patterns and the difficulty of inference. Moreover, they are often domain-specific (e.g., healthcare) and require significant human curation, hindering scalability and generalization.

Positioning Our Work:

Distinct from prior work, LUCARIO is constructed through a scalable, data-driven and human-validated process that enables probabilistic reasoning directly over tabular data. From real-world relational tables spanning diverse domains, Bayesian Networks are semi-automatically induced (not requiring manual structure design, while incorporating human knowledge and validation to ensure reliability and directional dependency). Each table is thus paired with its corresponding probabilistic graph. LUCARIO's scale allows the benchmark to capture a broader range of dependency patterns, including simple conditional relationships, multi-

	LUCARIO	QUITE	CLADDER	BLInD
Categorical vars.	✓	✓	✗	✗
Diverse domains	✓	✓	✓	✗
Diverse Q-types	✓	✗	✗	✗
Tabular data	✓	✗	✗	✗
Scalable BNs	✓	✗	✗	✗
Prob. insights	✓	✗	✗	✗

Table 1: Comparison of LUCARIO with recent benchmarks for probabilistic or causal reasoning.

variable interactions, and more complex dependency structures that rarely appear in small manually designed graphs. (Figure 1 shows a snippet of a graph structure).

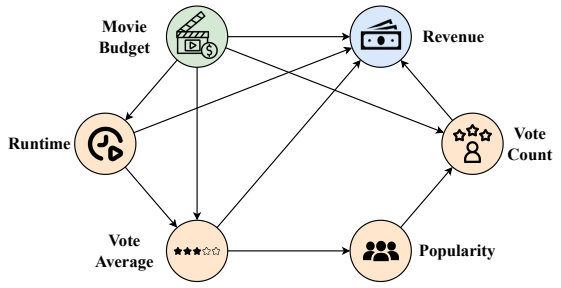


Figure 1: Expert Verified Data-Driven Bayesian Network Structure on Movie table

LUCARIO also provides premise sets describing inter-variable dependencies, and insights (reduced set of relevant premises – see § 3). Averaging 79k premises per table, these sets vastly exceed those in other benchmarks, offering a richer foundation for symbolic and neural reasoning under realistic, large-scale conditions where probabilistic and directional relationships arise organically from data.

Table 1 highlights the key differences between existing benchmarks and LUCARIO. To our knowledge, LUCARIO is the first benchmark to pair expert-validated Bayesian Networks, constructed semi-automatically from real, domain-diverse tables, with large-scale probabilistic QA evaluation.

3 LUCARIO Benchmark

We introduce LUCARIO, a benchmark built upon the BIRD dataset (Li et al., 2023), designed to evaluate probabilistic question answering (QA) over tabular data. While BIRD focuses on translating factual queries into SQL, our benchmark introduces probabilistic queries requiring reasoning under uncertainty. The benchmark comprises three primary components: relational tables, probabilistic queries, and ground-truth answers, supplemented by auxiliary reasoning artifacts (Bayesian Networks, premises, and insights).

3.1 Data Sources & Benchmark Artifacts

LUCARIO contains 10 real-world, large-scale relational databases from diverse domains (e.g., finance, healthcare, sports) sourced from BIRD and selected for meaningful probabilistic dependencies, feature richness, and statistical sufficiency (Appendix A.1). To support downstream neuro-symbolic reasoning, we pair each table with three auxiliary artifacts:

Bayesian Network (BN): A graphical model representing the directional statistical dependencies (Pearl, 1988) among the table's columns. A BN factorizes their joint distribution into local conditional probabilities, enabling exact computation of $P(X{=}x \mid \mathbf{C}{=}\mathbf{c})$ for any target and evidence.

Premises: Premises are a set of textual statements characterizing the statistical relationships inherent in the tabular data. These statements function as probabilistic rules, serving as a foundation for rule-based reasoning and explanation (Schrader et al., 2024). Formally, a single premise corresponds to a one-hop reasoning step that represents the conditional distribution of a target node given the specific states of its parent nodes.

Insights: A primary challenge of using premises as the core mechanism for the probabilistic reasoning, prediction, and explanation is the size of the premise set. Because it exhaustively covers all possible combinations of attribute states, the premise space grows exponentially for tables with ten or more attributes, introducing significant computational latency and complexity. To mitigate this, LUCARIO includes a set of insights—a compact subset of premises identified as having the most substantial impact on the data's behavior. We define the *Insight Saliency* of a variable X under a conditioning context C (the observed states of other columns, e.g., a customer's recent activity in the churn question of §1) as the Kullback–Leibler (KL) divergence between the conditional distribution $P_{X|C}$ and the base marginal distribution P_X:

$$\begin{aligned}\text{Saliency}(X, C) &= D_{\text{KL}}(P_{X|C} \parallel P_X) \\ &= \sum_{x \in X} P_{X|C}(x) \log \frac{P_{X|C}(x)}{P_X(x)} \quad (1)\end{aligned}$$

By systematically filtering for contexts C that exceed a minimum saliency threshold and introduce new probability peaks, we condense the massive

premise space into the most statistically significant conditional shifts, enabling efficient and highly calibrated reasoning.

3.2 Diversity in Queries and Questions

A key differentiator of LUCARIO compared to prior probabilistic reasoning benchmarks is the structural and linguistic diversity of its evaluation set. Because our semi-automatic construction pipeline (§3.3) is specifically engineered to generate these complex structures, we first define the target reasoning patterns and natural language variations that the benchmark encompasses.

Probabilistic Query Types (Reasoning Patterns). To enable systematic evaluation of probabilistic QA, LUCARIO includes a diverse set of probabilistic queries (Figure 2) covering distinct patterns of conditional dependence among table attributes: (i) **Forward Conditional Queries**: Probability of an outcome given observed conditions (e.g., the probability of high sales given that a marketing campaign was launched). (ii) **Diagnostic Queries**: Inferring the likelihood of an explanatory variable given observed evidence (e.g., the probability of a disease given a set of symptoms). (iii) **Interdependent Evidence Queries**: Multiple correlated variables provide alternative explanations for the same observation, such that evidence supporting one explanation reduces the probability of competing ones. (iv) **Multi-Evidence Aggregation Queries**: Handling scenarios with multiple pieces of evidence that may be overlapping or provide corroborative information.

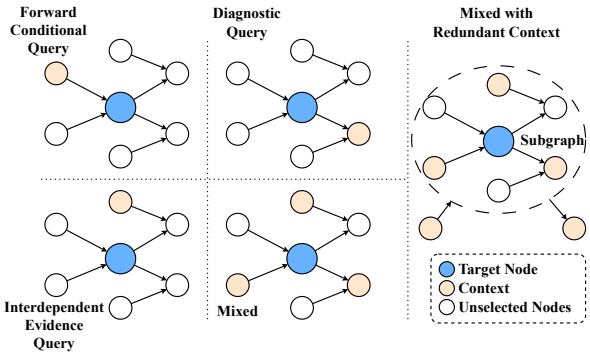


Figure 2: Probabilistic Query Types

Linguistic Question Variations. Real-world questions involving uncertainty are often expressed in diverse and nuanced ways. While many existing benchmarks for probabilistic reasoning primarily use formulations that closely mirror the symbolic representation of variables and states (e.g., a question contains "*temperature high*" when the graph node *temperature* includes a "*high*" state), real-world user queries frequently vary in specificity, granularity, and linguistic style.

To capture this variability and evaluate models' robustness under natural variation, LUCARIO expands each probabilistic query into four distinct linguistic expressions (Figure 3): (i) **Exact**: The question expresses the exact numeric range or symbolic state as defined in the BN. (ii) **Shifted**: The question refers to an alternative, valid numeric range. (iii) **Certain**: The condition is stated as a single numeric value instead of a range. (iv) **Natural**: The condition is described using free-form natural language paraphrasing.

Table 2 gives one example for each query type.

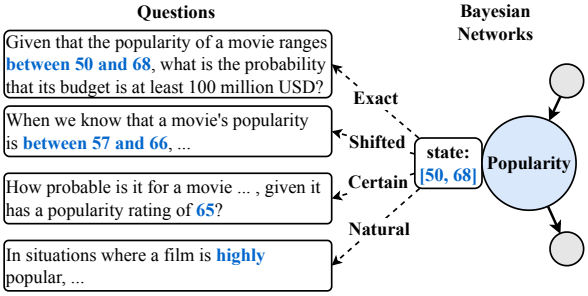


Figure 3: Question Types in LUCARIO

3.3 Dataset Creation

LUCARIO is constructed via a three-stage semi-automatic pipeline incorporating expert knowledge and validation.

Stage 1: BN Construction. For each selected table, continuous numerical columns are first discretized into semantically meaningful bins (e.g., mapping a raw *timestamp* to *morning* time range). We then apply the PC algorithm (Kalisch and Bühlman, 2007) to induce an initial Directed Acyclic Graph (DAG) representing the probabilistic dependencies between columns. To align the benchmark with domain logic and filter spurious statistical correlations, two domain experts manually review and refine these DAGs (adding, removing, or reversing edges), injecting semantic knowledge beyond what is observable in the raw data. When they disagreed or were uncertain about an edge, they consulted conditional-independence tests and the underlying records and resolved the issue through discussion. We subsequently fit Conditional Probability Distributions (CPDs) to these

Domain	Question	Variant	Gold
movies	*Given that a film has a moderately short duration and receives an above-average rating, how likely is it to generate low earnings?*	Natural	0.375
sales in weather	*When the average temperature is between 70 and 79 °F and the dew point is at least 66 °F, what is the probability that the calculated cooling degree is between 8 and 13 degree-days?*	Exact	0.567
hockey scoring	*Given that the number of goals scored is between 4 and 7, what is the probability that the shots on goal fall between 46 and 78?*	Shifted	0.535
cookbook nutrition	*Given that a recipe has between 0% to 5% of its calories coming from fat, 15% to 28.6% from protein, and 50% to 70% from carbohydrates, what is the probability that the cholesterol content in the recipe is between 0 to 6.53 milligrams?*	Exact	0.487
twitter	*Given that a tweet has received no likes, the sentiment analysis of the tweet is neutral, and it was posted on a Tuesday, what is the probability that the tweet was posted between 5 and 8 in the morning?*	Certain	0.266

Table 2: Example LUCARIO questions, one per probabilistic query type, from five domains and four linguistic variants.

validated DAGs using BDeu scoring over the table rows.

Following established protocols in probabilistic benchmarking (Jin et al., 2023; Schrader et al., 2024), the resulting expert-validated BN serves as the ground-truth structure, from which we automatically derive the full set of premises and a distilled set of high-impact insights.

Stage 2: Question Generation. Using the ground-truth BNs, we employ stratified sampling to select target variables in each BN. We traverse the graph to identify parent, child, and Markov-Blanket structures that align with our five defined probabilistic query reasoning patterns (Figure 2). We then prompt an LLM with the table descriptions, columns statistics, BN schema, selected anchor/-context nodes, and the target reasoning pattern to generate candidate natural language questions.

To ensure high linguistic quality and logical coherence, every candidate question undergoes manual screening for fluency, correctness, and contextual relevance; sub-optimal formulations are discarded. Finally, each accepted question is expanded into four distinct linguistic variants (Figure 3)—to evaluate model robustness against surface-level phrasing variations.

Stage 3: Answer Computation. To establish the ground-truth answers, we compute exact probabilities by performing inference directly on the expert-validated BN. For a given target variable X and conditioning context $\mathbf{C}$, the ground-truth probability is formally computed as $P(X{=}x \mid \mathbf{C}{=}\mathbf{c})$. Exact inference makes each answer analytically correct with respect to the expert-validated BN and its fitted CPDs, grounded in the table's data distribution, and fully reproducible.

3.4 Dataset Statistics

LUCARIO includes 10 datasets (tables), each with 157 probabilistic queries. Each query is expressed in four linguistic question variants, yielding 1,570 unique probabilistic scenarios and 6,280 question–answer pairs. The statistics are shown in Table 3, with more details in Appendix A.

Statistic	Mean Value
Number of Nodes per BN	12.50
Number of Edges per BN	32.60
States per Node	4.95
Total Degree (avg)	5.02
Number of Premises per BN	79,161.8
Number of Insights per BN	2,411.8
Premises per Node	6,043.9
Insights per Node	192.8

Table 3: Summary statistics of Bayesian Networks and probabilistic artifacts across benchmark tables.

3.5 Benchmark Quality Evaluation

To ensure the reliability of LUCARIO, we conducted both structural and linguistic evaluations of the benchmark components.

Graph Structure Evaluation. To evaluate the finalized benchmark artifacts, three researchers independent of the benchmark curation assessed the plausibility of a sampled subset of directional edges from the Ground-truth BNs. Provided with the graphs and underlying tabular data, the experts judged all sampled edges to be valid. Specifically, 84% of the edges were evaluated as intuitively consistent with domain knowledge, while the remaining 16% represented more subtle correlations that required inspection of the underlying data distributions to confirm.

Linguistic Quality. Following the protocol of Schrader et al. (2024), we evaluated the fluency of a random sample of 200 items (10 premises and 10 questions from each of the 10 tables) using Grammarly.[1] The corpus achieved an overall score of 96, significantly exceeding prior benchmarks (QUITE: 82–85, CLADDER: 45, BLInD: 34).

3.6 Evaluation Metrics

Given a natural language question on a table and its ground-truth probability, we evaluate model performance using the following metrics:

Error Rate: The percentage of predictions either outside the valid probability range $[0.0, 1.0]$ or missing entirely. A fallback value of $1.0/\text{num}_{\text{states}}$ is assigned, assuming a uniform prior. This generalizes the 0.5 fallback commonly used in existing binary-state benchmarks.

Mean Absolute Error (MAE): The average absolute difference between predicted and true probabilities, measuring overall estimation accuracy.

Root Mean Squared Error (RMSE): The square root of the average squared error, which penalizes large deviations more heavily than MAE.

Accuracy$_x$: The proportion of predictions within $\pm x$ of the ground-truth. We report $\text{Acc}_{0.02}$ and $\text{Acc}_{0.05}$ as thresholds for near-correct predictions.

4 Auto-BN: Probabilistic Inference over Tabular Data

While LLMs excel at factual tabular retrieval, they struggle with complex queries requiring reasoning under uncertainty. To bridge this gap, we propose **Auto-BN**, a fully automatic, end-to-end pipeline that induces BNs directly from raw tabular data and uses an LLM to translate natural language questions into exact probabilistic inference. Let $\mathcal{T}$ be a relational table and q a natural language question. The goal is to compute the grounded probability r:

$$r = P(\hat{n} = \hat{s} \mid C) = \frac{\sum_{\mathbf{u}} P(\hat{n}{=}\hat{s},\, C,\, \mathbf{u})}{\sum_{\hat{s}'} \sum_{\mathbf{u}} P(\hat{n}{=}\hat{s}',\, C,\, \mathbf{u})} \tag{2}$$

where $\hat{n}$ is the target attribute, $\hat{s}$ is its queried state, C is the conditioning context derived from q, and $\mathbf{u}$ represents the unobserved variables.

Auto-BN decomposes this challenge into a three-stage neuro-symbolic pipeline:

$$\mathcal{T} \xrightarrow{\S 4.1} \mathcal{B} \xrightarrow{\S 4.2} q' \xrightarrow{\S 4.3} r$$

where $\mathcal{B}$ is the automatically induced Bayesian Network, and $q' = (\hat{n},\, \hat{s},\, C)$ is the structured, executable query parsed from natural language.

4.1 Automatic BN Construction (Offline)

In contrast to the expert-involved Gold BNs used for benchmark evaluation, Auto-BN induces a data-driven BN $\mathcal{B}$ from the raw table $\mathcal{T}$ via a fully automated, four-step offline pipeline: (i) LLM-assisted column classification (binary, categorical, or numerical) based on value distributions, data types, and schema descriptions; (ii) K-means discretization of continuous columns, utilizing an LLM to assign semantically meaningful boundary labels; (iii) directed dependency graph induction via the PC algorithm, with algorithmic post-processing to enforce acyclicity and prune low-confidence edges; and (iv) Conditional Probability Table (CPT) fitting via Bayesian estimation directly over the tabular records. Discretization lets parameter learning and exact inference use finite-state CPTs without the distributional assumptions (e.g., linear-Gaussian) required by continuous or hybrid BNs. It also lets one BN handle categorical and numeric columns uniformly. Executed once per table, this pipeline yields a purely data-driven probabilistic model. By decoupling the induced graph from the human-verified ground-truth BNs, we ensure that subsequent evaluations rely exclusively on original data, thereby precluding any ground-truth leakage.

4.2 Robust Query Translation

At inference time, an LLM parses q into a structured node-mapping query $q' = \langle \hat{n}, \hat{s}, C \rangle$. Drawing inspiration from execution-guided Text-to-SQL pipelines (Pourreza et al., 2024; Talaei et al., 2024), Auto-BN mitigates single-pass schema-linking errors via a three-phase fault-tolerant mechanism:

Multi-Path Cross-Consistency Voting. Auto-BN concurrently samples N (default: 3) diverse parse candidates. Each is validated against the BN schema to ensure all nodes and states strictly exist in BN $\mathcal{B}$. The plurality tuple among the *valid* candidates is selected, surfacing semantic ambiguity as structured uncertainty rather than a silent failure.

Execution-Guided Iterative Refinement. If all N candidates fail validation, Auto-BN extracts the specific error trace (e.g., invalid target node) from a fresh, zero-temperature parse. This execution feedback is appended to the prompt, allowing the LLM to self-correct its schema linking for up to K_{corr} rounds (default: $K_{\text{corr}} = 2$), recovering the

[1] https://www.grammarly.com

vast majority of initial vocabulary mismatches.

Soft Fallback and Typed Error Taxonomy. Upon depleting the correction budget, Auto-BN assigns a typed error (e.g., `invalid_state`). If the initial voting phase produced any parseable candidate, this best-effort mapping is forwarded to the inference fallback chain (§4.3) instead of being discarded, maximizing structural recovery before abstention. Together, these mechanisms reduce the parser error rate from 5.6–20.8% under single-pass parsing to near zero across three LLM backbones (Appendix D.6, Table 14).

4.3 Inference with Multi-Level Relaxation

A structurally valid query q' is executed via variable elimination (Eq. 2). To handle queries suffering from persistent vocabulary mismatches or execution-time logical exceptions (including those forwarded via the soft fallback mechanism), Auto-BN applies a four-level heuristic relaxation chain: (i) *Semantic Node Relaxation* replaces an unmapped $\hat{n}$ with the closest valid BN node via cosine similarity using sentence embeddings; (ii) *Semantic State Relaxation* similarly maps invalid states $\hat{s}$ or states within C to their nearest valid neighbors; (iii) *Context Pruning* iteratively strips conflicting evidence variables from C; and (iv) *Marginal Fallback* returns the unconditioned prior $P(\hat{n} = \hat{s})$ if all conditioned variants fail. Appendix F describes Auto-BN-Tool, an agentic tool-calling extension of this pipeline, and Appendix G provides the full prompt templates.

5 Experimental Evaluation

We evaluate the Auto-BN framework and baselines on the 10 datasets in LUCARIO benchmark.

5.1 Baselines

We compare our framework against a diverse set of baselines that reflect factual and probabilistic approaches to probabilistic QA over tabular data:

NL2SQL: Apply recent NL2SQL work (Gao et al., 2023) in probabilistic setting, where the model generates a SQL query from the question, executes it on the database, and uses the result to infer a probabilistic answer.

LLM + Table: A direct prompting strategy where the question is answered using the table description and a random sample of rows (until reaching the maximum context window size), without structured reasoning or symbolic modeling.

Reasoning Models: Same strategy as LLM + Table, but using advanced reasoning models and their inherently internal CoT.

TableRAG: TableRAG (Chen et al., 2024) is a large table understanding approach by LLM with retrieval-augmented generation strategy.

LLM + Premise: This baseline resembles Causal-CoT (Jin et al., 2023) extended with large-scale probabilistic premise retrieval. It identifies nodes and states from the question using LLM, retrieves probabilistic premises, then uses LLM to answer the question with all information. Three premise retrieval strategies (Appendix B) are evaluated.

LLM + Premise + Insights: Extends the previous approach by also incorporating a small set (20) of high-impact insights generated by our proposed Markov-Blanket-based insight generation approach (see Appendix C for details).

Auto-BN: Our proposed fully automated neuro-symbolic pipeline (§4). To ensure a strict, real-world evaluation, Auto-BN does not access the human-verified Gold BNs, premises, or insights provided by the benchmark.

Table 9 summarizes the benchmark artifacts available to each baseline. Complete implementation details for all baselines, including prompt templates, premise storage formats, and retrieval strategies (Vector, BM25, Hybrid), are in Appendix C. Appendix E replaces the induced BN with the expert-validated Gold BN (83.8% vs. 46.7% $\text{Acc}_{0.05}$ with GPT-4o) and shows low directed overlap between the two (F1 of 0.159).

5.2 Main Results

Table 4 reports the performance of all evaluated methods across various configurations[2]. We report the mean and standard deviation of each metric across all benchmark datasets (tables), computed per (question type, baseline) pair. The standard deviation reflects performance variability across 10 datasets for the same question type and method. Our results highlight several key observations:

Factual QA Methods. While LLMs prompted with raw table rows (LLM+Table) achieve reasonably low MAE and RMSE, their accuracy in predicting precise probabilities (Acc_x) is limited. These models rely on memorization and pattern-matching from examples but lack the structured grounding necessary for calibrated inference.

[2]Each result corresponds to a single run per method per dataset. Reported mean and standard deviation are computed across benchmark datasets for each question type and method.

Type	Method	LLM	% $\text{Acc}_{0.02}$ ↑	% $\text{Acc}_{0.05}$ ↑	MAE ↓	RMSE ↓	% Error Rate ↓
	Random	-	4.5 ± 1.8	11.1 ± 2.7	0.362 ± 0.021	0.447 ± 0.023	0.0 ± 0.0
Factual	LLM + Table	GPT-4o	11.1 ± 6.8	24.4 ± 9.6	0.171 ± 0.046	0.237 ± 0.060	0.0 ± 0.0
		Llama-3.3-70B	11.1 ± 5.1	25.4 ± 8.1	0.203 ± 0.041	0.284 ± 0.052	0.0 ± 0.0
		Mixtral-8x22B	10.8 ± 4.3	23.4 ± 8.2	0.179 ± 0.042	0.243 ± 0.051	0.0 ± 0.0
	Reasoning Models	GPT-5.4	11.4 ± 4.6	25.3 ± 7.9	0.199 ± 0.067	0.278 ± 0.074	0.0 ± 0.0
		DeepSeek-R1	11.4 ± 6.9	21.9 ± 12.8	0.204 ± 0.051	0.276 ± 0.064	0.0 ± 0.0
	TableRAG	GPT-4o	7.6 ± 5.0	18.6 ± 7.6	0.220 ± 0.037	0.297 ± 0.046	0.0 ± 0.0
		Llama-3.3-70B	7.4 ± 4.9	19.4 ± 7.3	0.226 ± 0.052	0.303 ± 0.062	0.0 ± 0.0
		Mixtral-8x22B	6.6 ± 4.1	18.7 ± 8.1	0.217 ± 0.046	0.285 ± 0.052	0.0 ± 0.0
	NL2SQL	GPT-4o	13.2 ± 9.7	25.1 ± 14.5	0.190 ± 0.062	0.260 ± 0.075	6.2 ± 7.9
		Llama-3.3-70B	14.6 ± 9.9	25.8 ± 13.8	0.201 ± 0.067	0.278 ± 0.082	3.2 ± 6.2
		Mixtral-8x22B	7.3 ± 5.7	17.3 ± 7.4	0.186 ± 0.044	0.253 ± 0.063	52.8 ± 30.3
Prob. Premises	LLM+Premise(Vector)	GPT-4o	8.4 ± 4.4	20.5 ± 6.3	0.178 ± 0.047	0.243 ± 0.062	0.0 ± 0.0
		Llama-3.3-70B	6.7 ± 4.8	16.8 ± 7.4	0.227 ± 0.050	0.287 ± 0.056	0.1 ± 0.4
		Mixtral-8x22B	4.8 ± 2.8	14.2 ± 4.7	0.299 ± 0.044	0.404 ± 0.048	2.6 ± 2.3
	LLM+Premise(BM25)	GPT-4o	8.7 ± 5.8	19.3 ± 8.0	0.184 ± 0.029	0.253 ± 0.036	0.0 ± 0.0
		Llama-3.3-70B	6.9 ± 4.2	17.0 ± 6.9	0.235 ± 0.048	0.298 ± 0.052	0.1 ± 0.4
		Mixtral-8x22B	6.1 ± 3.7	15.0 ± 6.2	0.310 ± 0.039	0.416 ± 0.042	1.9 ± 1.5
	LLM+Premise(Hybrid)	GPT-4o	10.2 ± 5.9	23.1 ± 8.4	0.163 ± 0.032	0.228 ± 0.043	0.0 ± 0.0
		Llama-3.3-70B	9.2 ± 5.1	20.6 ± 7.8	0.212 ± 0.045	0.273 ± 0.052	0.1 ± 0.4
		Mixtral-8x22B	6.6 ± 4.1	15.9 ± 6.4	0.275 ± 0.043	0.378 ± 0.053	2.5 ± 2.3
Prob. Insights	LLM+Premise(Vector) +Insights	GPT-4o	24.7 ± 11.4	39.9 ± 12.1	0.128 ± 0.042	0.198 ± 0.054	0.0 ± 0.0
		Llama-3.3-70B	10.7 ± 6.3	24.6 ± 10.6	0.205 ± 0.047	0.275 ± 0.052	0.1 ± 0.4
		Mixtral-8x22B	10.2 ± 4.5	21.5 ± 5.9	0.277 ± 0.056	0.395 ± 0.062	1.0 ± 1.5
	LLM+Premise(BM25) +Insights	GPT-4o	23.3 ± 10.1	37.6 ± 11.5	0.132 ± 0.034	0.203 ± 0.046	0.0 ± 0.0
		Llama-3.3-70B	11.3 ± 6.7	24.1 ± 11.5	0.203 ± 0.044	0.272 ± 0.047	0.1 ± 0.4
		Mixtral-8x22B	10.2 ± 4.9	20.9 ± 7.9	0.290 ± 0.054	0.410 ± 0.059	0.8 ± 1.2
	LLM+Premise(Hybrid) +Insights	GPT-4o	23.9 ± 10.7	39.2 ± 11.9	0.127 ± 0.037	0.197 ± 0.049	0.0 ± 0.0
		Llama-3.3-70B	11.6 ± 6.3	25.0 ± 10.9	0.201 ± 0.046	0.267 ± 0.053	0.1 ± 0.4
		Mixtral-8x22B	10.9 ± 5.3	23.4 ± 7.2	0.255 ± 0.046	0.373 ± 0.049	0.8 ± 1.4
	Auto-BN	GPT-4o	$\mathbf{28.0 \pm 15.5}$	$\mathbf{46.7 \pm 16.2}$	$\mathbf{0.107 \pm 0.043}$	0.169 ± 0.058	0.0 ± 0.0
		Llama-3.3-70B	25.4 ± 13.6	44.8 ± 15.0	0.109 ± 0.040	$\mathbf{0.168 \pm 0.053}$	0.0 ± 0.0
		Mixtral-8x22B	26.1 ± 15.4	44.6 ± 16.1	0.112 ± 0.046	0.174 ± 0.062	0.0 ± 0.2

Table 4: Evaluation across factual and probabilistic reasoning strategies across different LLMs.

Moreover, this performance is achieved using extra-long contexts, making such approaches computationally expensive and impractical at scale. Even when utilizing advanced reasoning models (GPT-5.4, DeepSeek-R1), direct probabilistic derivation based on raw tables remains unsatisfactory, indicating that internal multi-step reasoning cannot easily substitute for exact statistical aggregation.

Despite being designed for large-table understanding, TableRAG underperforms even the basic LLM+Table baseline (7.6% vs. 11.1% $\text{Acc}_{0.02}$ with GPT-4o). This exposes a fundamental mismatch: TableRAG optimizes for localized fact lookup ("needles in a haystack"). By retrieving a bounded set of discrete cell values without their underlying frequency statistics, it cannot aggregate the global evidence required to reconstruct the dense marginal and conditional distributions necessary for probabilistic reasoning.

NL2SQL-style models perform poorly in this setting, as expected. They frequently fail to produce valid SQL under zero-shot prompting, especially Mixtral. More fundamentally, deterministic query execution cannot express conditional uncertainty. For instance, retrieving zero matched records does not equate to a zero probability—highlighting a core limitation of factual QA pipelines when applied to probabilistic tasks.

Premise-Based Inference: Inspired by approaches like CLADDER that use LLMs for one-hop probabilistic reasoning, we evaluated whether our benchmark could support similar derivations. However, due to the significantly larger and more complex premise sets in LUCARIO (79k premises per table), LLMs struggled to combine the relevant evidence into coherent probabilistic derivations. Among premise retrievers, vector-based retrieval consistently outperformed BM25. Hybrid retrieval (combining both methods) yields marginal improvement over individual retrieval.

Incorporating Insights: Across all retrieval strategies, augmenting retrieved premises with a compact set of 20 high-impact insights substantially improved performance. By distilling the large premise space into a concise and informative subset, insights reduce cognitive overhead, improve

answer calibration, and mitigate hallucinations in downstream reasoning.

Auto-BN: Auto-BN substantially outperforms all baselines across core metrics. With GPT-4o, it achieves 28.0% $\text{Acc}_{0.02}$ and MAE of 0.107, while successfully maintaining a near-zero error rate. This demonstrates the effectiveness of integrating probabilistic graphical model reasoning with robust LLM-based query understanding. By learning BNs directly from data and executing fail-safe semantic parsing, Auto-BN provides highly accurate answers grounded in exact probabilistic inference.

Agentic Extension (Auto-BN-Tool). To somehow relax the limitations of the pipelined approach and allow LLM-based self recovery, we extend **Auto-BN** into a callable tool for multi-turn LLM agents. This formulation enables the LLM to autonomously invoke probability calculations, iteratively decompose compositional queries, and synthesize comprehensive answers from multiple probabilistic sub-steps. We ran Auto-BN-Tool with GPT-4o and its performance is very close to Auto-BN (more details in Appendix F).

Probability Hallucination vs. Grounded Evidence: While LLM baselines artificially achieve near-zero error rates by forcing formatted outputs, they frequently hallucinate uniform priors when uncertain (e.g., predicting 0.2 for 5 candidate states). Auto-BN strictly precludes this by grounding every prediction in exact Bayesian inference. Its own near-zero error rate stems from robust query translation, execution-guided refinement and semantic relaxation rather than blind guessing, guaranteeing a transparent, verifiable reasoning trace.

Further experimental results, including performance breakdowns, running time, multi-hop premises and ablation study are in Appendix D.

5.3 Effect of Number of Premises

To study the effect of premise count, we conduct experiments on the complex *hockey scoring* dataset using the LLM + Premises setup. As shown in Figure 4, increasing the number of retrieved premises yields marginal improvements in MAE and RMSE, but does not consistently boost accuracy. This suggests diminishing returns, where additional premises may introduce redundancy or noise. In contrast, supplementing with a small set of high-impact insights yields in more substantial performance gains, as shown in Table 4.

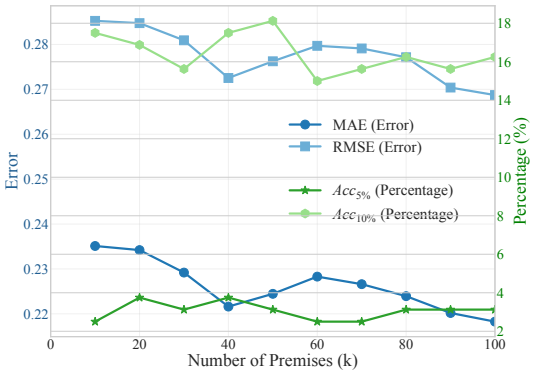


Figure 4: Impact of # premises on performance

6 Conclusion

We introduce LUCARIO, a benchmark for probabilistic question answering over real-world tabular data, and Auto-BN, a framework for answering such questions. Unlike prior work that relies on small, curated Bayesian Networks or synthetic templates, LUCARIO supports large-scale, data-driven reasoning by inducing and human-validating Bayesian Networks from relational tables. It includes diverse query types, natural language variations, and structured artifacts such as premises and insights. Auto-BN combines probabilistic inference with robust LLM-based query translation. Experimental results show that our framework outperforms retrieval-based and probability-based baselines, highlighting the value of integrating probabilistic graphical models with neural models for reasoning under uncertainty.

Limitations

While our framework effectively learns Bayesian network structures from tabular records, it operates in a purely data-driven manner and does not incorporate external knowledge or semantic priors beyond human validation of the benchmark artifacts. Consequently, the learned statistical dependencies are limited to patterns present in the observed data.

We emphasize that our framework evaluates *observational probabilistic reasoning*. While we occasionally employ terminology from the causal discovery literature (e.g., the PC algorithm) to compare with related work, we do not claim the discovery of true causal mechanisms or the estimation of interventional effects (do-calculus). Some directed edges in the induced Bayesian Networks represent strict statistical correlations (Markov Equivalence) rather than verified causal mechanisms.

Finally, while we introduced a new benchmark

and demonstrated strong baseline methods, we do not claim our proposed Auto-BN pipeline to be the optimal solution for this benchmark. The benchmark is intended as a platform to foster further exploration at the intersection of tabular reasoning, uncertainty quantification, and probabilistic modeling.

Ethical Considerations

Our framework automatically fit given tabular data across domains, which may propagate underlying biases in the data. The learned structures reflect statistical estimations based solely on observed records, without incorporating broader world knowledge. As such, they may not represent ground truth and should not be used as generic justification or reliable decision making, such as investigations or healthcare applications.

Acknowledgments

The authors thank Sajjadur Rahman for his contributions to an earlier version of this work. As he was not involved in the substantial revisions and preparation of the present manuscript, he chose not to be included as an author of this version.

References

Ankur Ankan and Johannes Textor. 2024. pgmpy: A python toolkit for bayesian networks. *Journal of Machine Learning Research*, 25(265):1–8.

Alexander Bondarenko, Magdalena Wolska, Stefan Heindorf, Lukas Blübaum, Axel-Cyrille Ngonga Ngomo, Benno Stein, Pavel Braslavski, Matthias Hagen, and Martin Potthast. 2022. Causalqa: A benchmark for causal question answering. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 3296–3308.

Neeloy Chakraborty, Melkior Ornik, and Katherine Driggs-Campbell. 2025. Hallucination detection in foundation models for decision-making: A flexible definition and review of the state of the art. *ACM Computing Surveys*.

Si-An Chen, Lesly Miculicich, Julian Martin Eisenschlos, Zifeng Wang, Zilong Wang, Yanfei Chen, Yasuhisa Fujii, Hsuan-Tien Lin, Chen-Yu Lee, and Tomas Pfister. 2024. Tablerag: Million-token table understanding with language models. In *Advances in Neural Information Processing Systems*, volume 37.

Luc De Raedt, Angelika Kimmig, and Hannu Toivonen. 2007. Problog: A probabilistic prolog and its application in link discovery. In *IJCAI 2007, Proceedings of the 20th international joint conference on artificial intelligence*, pages 2462–2467. IJCAI-INT JOINT CONF ARTIF INTELL.

Xiang Deng, Huan Sun, Alyssa Lees, You Wu, and Cong Yu. 2022. Turl: Table understanding through representation learning. *ACM SIGMOD Record*, 51(1):33–40.

Dawei Gao, Haibin Wang, Yaliang Li, Xiuyu Sun, Yichen Qian, Bolin Ding, and Jingren Zhou. 2023. Text-to-sql empowered by large language models: A benchmark evaluation. *arXiv preprint arXiv:2308.15363*.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, and 1 others. 2024. The Llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Jonathan Herzig, Pawel Krzysztof Nowak, Thomas Mueller, Francesco Piccinno, and Julian Eisenschlos. 2020. Tapas: Weakly supervised table parsing via pre-training. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4320–4333.

Lei Huang, Weijiang Yu, Weitao Ma, Weihong Zhong, Zhangyin Feng, Haotian Wang, Qianglong Chen, Weihua Peng, Xiaocheng Feng, Bing Qin, and 1 others. 2025. A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions. *ACM Transactions on Information Systems*, 43(2):1–55.

Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, and 1 others. 2024. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*.

Albert Q Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, and 1 others. 2024. Mixtral of experts. *arXiv preprint arXiv:2401.04088*.

Zhijing Jin, Yuen Chen, Felix Leeb, Luigi Gresele, Ojasv Kamal, Zhiheng Lyu, Kevin Blin, Fernando Gonzalez Adauto, Max Kleiman-Weiner, Mrinmaya Sachan, and 1 others. 2023. Cladder: Assessing causal reasoning in language models. *Advances in Neural Information Processing Systems*, 36:31038–31065.

Markus Kalisch and Peter Bühlman. 2007. Estimating high-dimensional directed acyclic graphs with the pc-algorithm. *Journal of Machine Learning Research*, 8(3).

Boyan Li, Yuyu Luo, Chengliang Chai, Guoliang Li, and Nan Tang. 2024. The dawn of natural language to sql: are we fully ready? *arXiv preprint arXiv:2406.01265*.

Ce Li, Xiaofan Liu, Zhiyan Song, Ce Chi, Boshen Shi, Chen Zhao, Guanguang Chang, Zhendong Wang, Kexin Yang, Xing Wang, Chao Deng, and Junlan Feng. 2026. TReB: A comprehensive benchmark for evaluating table reasoning capabilities of large language models. In *Proceedings of the 49th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 3267–3275. Association for Computing Machinery.

Jinyang Li, Binyuan Hui, Ge Qu, Jiaxi Yang, Binhua Li, Bowen Li, Bailin Wang, Bowen Qin, Ruiying Geng, Nan Huo, and 1 others. 2023. Can LLM already serve as a database interface? A BIg bench for large-scale database grounded text-to-SQLs. In *Advances in Neural Information Processing Systems*, volume 36, pages 42330–42357.

Jerry Liu. 2022. LlamaIndex. `https://github.com/run-llama/llama_index`. Software.

Xiao Liu, Zirui Wu, Xueqing Wu, Pan Lu, Kai-Wei Chang, and Yansong Feng. 2024. Are llms capable of data-based statistical and causal reasoning? benchmarking advanced quantitative reasoning with data. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 9215–9235.

Meta. 2024. Llama 3.3 model card. `https://github.com/meta-llama/llama-models/blob/main/models/llama3_3/MODEL_CARD.md`.

Mistral AI. 2024. Cheaper, better, faster, stronger. `https://mistral.ai/news/mixtral-8x22b`. Blog post announcing Mixtral 8x22B, 17 April 2024.

Aliakbar Nafar, Kristen Brent Venable, and Parisa Kordjamshidi. 2025. Reasoning over uncertain text by generative large language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 24911–24920.

Judea Pearl. 1988. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

Mohammadreza Pourreza, Hailong Li, Ruoxi Sun, Yeounoh Chung, Shayan Talaei, Gaurav Tarlok Kakkar, Yu Gan, Amin Saberi, Fatma Ozcan, and Sercan O Arik. 2024. Chase-sql: Multi-path reasoning and preference optimized candidate selection in text-to-sql. *arXiv preprint arXiv:2410.01943*.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Timo Pierre Schrader, Lukas Lange, Simon Razniewski, and Annemarie Friedrich. 2024. QUITE: Quantifying uncertainty in natural language text in Bayesian reasoning scenarios. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 2634–2652, Miami, Florida, USA. Association for Computational Linguistics.

Chen Shen, Jin Wang, Sajjadur Rahman, and Eser Kandogan. 2024. Demonstration of a multi-agent framework for text to sql applications with large language models. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*, CIKM '24, page 5280–5283, New York, NY, USA. Association for Computing Machinery.

Safal Shrestha, Minwu Kim, and Keith Ross. 2025. Mathematical reasoning in large language models: Assessing logical and arithmetic errors across wide numerical ranges. *arXiv preprint arXiv:2502.08680*.

Aarohi Srivastava, Abhinav Rastogi, Abhishek Rao, and 1 others. 2023. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *Transactions on Machine Learning Research*. Featured Certification.

Shayan Talaei, Mohammadreza Pourreza, Yu-Chen Chang, Azalia Mirhoseini, and Amin Saberi. 2024. Chess: Contextual harnessing for efficient sql synthesis. *arXiv preprint arXiv:2405.16755*.

Xianjie Wu, Jian Yang, Linzheng Chai, Ge Zhang, Jiaheng Liu, Xeron Du, Di Liang, Daixin Shu, Xianfu Cheng, Tianzhen Sun, Tongliang Li, Zhoujun Li, and Guanglin Niu. 2025. TableBench: A comprehensive and complex benchmark for table question answering. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 25497–25506.

Jianbin Zhao, Pengfei Zhang, Yuzhen Wang, Rui Xin, Xiuyuan Lu, Ripeng Li, Shuai Lyu, Zhonghong Ou, and Meina Song. 2025. Tqagent: Enhancing table-based question answering with knowledge graphs and tree-structured reasoning. *Applied Sciences (2076-3417)*, 15(7).

A Benchmark Details

We provide detailed statistics for the LUCARIO benchmark, covering data source selection, the structural properties of the Bayesian Networks, and the distribution of questions across reasoning patterns.

A.1 Data Source Selection

LUCARIO is built on 10 real-world relational tables drawn from the BIRD benchmark (Li et al., 2023). These 10 tables were carefully selected from hundreds of candidate sources in BIRD according to three criteria that are necessary for constructing a meaningful probabilistic QA benchmark:

Meaningful Probabilistic Dependencies: The schema must exhibit logical, directional associations between attributes (e.g., *departure delay* influencing *arrival delay*), enabling non-trivial conditional reasoning.

Feature Richness: The table must feature a high density of categorical or discretizable numerical columns. Databases consisting primarily of unique identifiers, proper names, or unstructured locations were excluded.

Statistical Sufficiency: The table must contain a sufficient volume of records to ensure that empirical probability distributions and the induced statistical models (e.g., Bayesian Networks) are robust, reliable, and representative of real-world complexity.

The resulting 10 tables span diverse real-world domains, from transportation and finance to sports and social media. Table 5 summarizes the key properties of each source table.

A.2 Detailed Statistics

Bayesian Network Properties. As shown in Table 6, the Bayesian Networks induced in LUCARIO are substantially larger and more structured than those used in prior benchmarks. On average, each network contains over 12 nodes, with approximately 6,000 probabilistic premises per node. This scale highlights a key challenge: methods that require injecting all premises into LLM prompts, such as CLADDER-style symbolic chains, are fundamentally infeasible due to context length limits and reasoning bottlenecks. This motivates our design of insight-driven filtering and structured inference via Auto-BN (§ 4).

Question Distribution. Table 7 summarizes the distribution of question types across five probabilistic query types. For each probabilistic query, four variations in question formulation are created: *Exact*, *Shifted*, *Certain*, and *Natural*, yielding an equal number of questions per type (table rows). The probabilistic reasoning query types include *forward conditional query* and *diagnostic query*, as well as more complex reasoning patterns such as *interdependent evidence query* and *mixed* queries, with or without redundant context. The benchmark is also balanced across query types (table columns), ensuring broad coverage and structural diversity.

Evaluation Splits. Following the protocol of zero-shot benchmarks such as BIG-Bench (Srivastava et al., 2023), we treat each table as an independent evaluation unit, with no train/test split. This setup assesses inherent reasoning capabilities without fine-tuning. An optional train/test split is provided for users interested in training/fine-tuning based experiments.

Privacy and Licensing. The LUCARIO benchmark does not include names or any information that uniquely identifies individuals. In accordance with its data sources, the benchmark is released under the CC BY-SA 4.0 license.

A.3 Examples by Probabilistic Query Type

Table 8 shows one *Natural*-variant question for each probabilistic query type from the *movies* table.

A.4 Worked Example

This example traces one LUCARIO instance from the input table to the gold answer.

Input table. The *movies* table has 4,627 rows and 13 columns (Table 5), including runtime, vote_average, vote_count, and popularity; a typical row has runtime 105, vote_average 7.3, vote_count 959, and popularity 23.7.

BN. Figure 1 shows the Gold BN, whose nodes are the discretized columns.

Question. *"Given that a movie has a runtime of 103 minutes and an average vote of 8.9, what is the probability that its vote count is between 750 and 3000?"* (*Certain* variant).

BN query. $P(\textit{vote_count} \in (750, 3000] \mid \textit{runtime} \in (100, 110], \textit{vote_average} \in (7.0, 10.0])$: each stated value is mapped to the bin that contains it.

Inference path. Exact inference propagates the evidence over two hops, vote_average $\rightarrow$ popularity $\rightarrow$ vote_count.

Answer. The probability rises from the prior $P(\textit{vote_count} \in (750, 3000]) = 0.209$ to the gold answer of 0.352.

B Premise Store and Retrieval

To enable retrieval on large scale premises, for each premise, LUCARIO provides two types of premises stores: the Numeric Premise Store stores the nodes and states in numeric format (e.g. "*price increases from 1% to 5%*"), and the Natural Premise Store stores in natural language text format ("*price slightly increases*").

We designed 3 types of premise store retrievers base on LlamaIndex (Liu, 2022):

- **Vector** Retrieve premises according to text encoding SentenceTransformer (Reimers and Gurevych, 2019) similarity.
- **BM25** Retrieve premises using the BM25 (Best Match 25) method that effectively ranks

Table	Rows	Columns	Domain	Notes
airlines	701,352	28	Transportation	Real-world flight data including delays and routes
basketball players	23,751	43	Sports	Season-level performance statistics for NBA & ABA players
bike stations weather	3,665	24	Urban Mobility	Bike share stations & weather data
coinmarketcap	4,441,972	19	Finance	Real-world crypto market history
college completion	3,798	62	Education	US college graduation statistics from IPEDS
cookbook nutrition	878	16	Health	Nutritional info from online recipe collections
hockey scoring	45,967	31	Sports	Professional hockey players statistics per season
movies	4,627	13	Entertainment	Metadata from popular movie databases
sales in weather	20,517	20	Retail	Sales transactions combined with weather info
twitter	99,901	14	Social Media	Tweet metadata sampled over multiple months
Mean	**486,061**	**25.91**	–	–

Table 5: Summary of tables used in experiments

Dataset	Nodes	States/Node	Premises	Insights	Prem/Node	Ins/Node	Edges	In-Deg	Out-Deg	Deg
airlines	7	5.00	5,107	130	729.6	18.6	10	1.4	1.4	2.9
basketball players	19	5.00	15,568	5,652	819.4	297.5	42	2.2	2.2	4.4
bike stations weather	17	4.94	23,217	1,168	1,365.7	68.7	43	2.5	2.5	5.1
coinmarketcap	13	5.00	43,932	960	3,379.4	73.8	37	2.9	2.9	5.7
college completion	15	4.67	46,337	733	3,089.1	48.9	48	3.2	3.2	6.4
cookbook nutrition	14	5.00	4,798	4,153	342.7	296.6	28	2.0	2.0	4.0
hockey scoring	13	4.92	337,559	3,063	25,966.1	235.6	41	3.2	3.2	6.3
movies	6	5.00	669	1,634	111.5	272.3	10	1.7	1.7	3.3
sales in weather	13	5.00	305,100	4,405	23,469.2	338.9	48	3.7	3.7	7.4
twitter	8	4.88	9,331	2,220	1,166.4	277.5	19	2.4	2.4	4.8
Mean	12.5	4.95	79,161.8	2,411.8	6,043.9	192.8	32.6	2.5	2.5	5.0

Table 6: Statistics of Bayesian Networks for each dataset in LUCARIO. Degree metrics are averaged per node.

Question Type	Probabilistic Query Types					Total
	Forward Conditional	Diagnostic	Interdependent Evidence	Mixed	Mixed with Redundant Context	
Exact	326	320	304	315	305	1570
Shifted	326	320	304	315	305	1570
Certain	326	320	304	315	305	1570
Natural	326	320	304	315	305	1570
Total	1304	1280	1216	1260	1220	6280

Table 7: Distribution of question types across probabilistic query types (reasoning patterns).

premises based on query term occurrence and rarity across the corpus.

- **Hybrid** Retrieve from both Vector and BM25 Retriever, combine and rerank the results.

C Baselines Implementation Details

We compare our Auto-BN framework against a diverse set of baselines spanning LLM-only, retrieval-augmented, and symbolic approaches. Below we describe the artifacts summary and implementation details referenced in § 5.

C.1 Artifacts Summary

Table 9 summarizes the benchmark artifacts used by each method, making the information access conditions explicit for direct comparison. No baselines utilize the ground-truth BN.

Query Type	Question
Forward Conditional	*If we observe that a movie has a short duration and receives below average ratings, how likely is it that the film will generate little to no revenue?*
Diagnostic	*Given that the movie has received only a small amount of votes, how likely is it that its popularity is very low?*
Interdependent Evidence	*Given that the movie has an extremely high budget, how likely is it to achieve moderate popularity?*
Mixed	*Given that a movie has an extended duration, receives a moderate amount of votes, and is produced with a very limited budget, how likely is it that this movie achieves a high average rating?*
Mixed with Redundant Context	*Given that a movie has a moderately short duration, receives high ratings, but generates little to no revenue, how likely is it that the movie will have very low popularity?*

Table 8: One *Natural*-variant question for each probabilistic query type (*movies*).

Method	Raw Table	Ground-Truth BN	Premises	Insights
LLM + Table	✓			
NL2SQL	✓			
TableRAG	✓			
Reasoning Model	✓			
LLM+Premise			✓	
LLM+Premise+Insights			✓	✓
Auto-BN	✓			
*Auto-BN (Gold)**	✓	✓		

Table 9: Benchmark artifacts used by each method. All baselines utilize the table description to understand the natural language question. Premises and Insights are atomic minimum one-hop reasoning instances derived from the human-verified Ground-Truth BN provided by the benchmark. Auto-BN relies exclusively on the Raw Table to induce its own probabilistic model. (**Auto-BN (Gold)* is an upper-bound ablation evaluated in Appendix E to isolate query translation performance).

C.2 Auto-BN Evaluation

While the technical mechanisms of the Auto-BN pipeline are detailed in §4, we emphasize a critical aspect of its evaluation protocol here. In our main experiments, Auto-BN operates in a strictly automated, *structure-free* setting. It accesses only the raw tables and their descriptions, without any access to the ground-truth BNs. Consequently, the data-specific Bayesian Networks induced by Auto-BN act purely as observational probabilistic representations of the underlying tabular data, and do not benefit from the domain experts' manual refinement or injection of semantic knowledge.

By contrast, the premise- and insight-augmented baselines leverage artifacts derived directly from the expert-curated ground-truth BNs. We intentionally adopt this asymmetric evaluation protocol to prevent structural data leakage and rigorously test Auto-BN's fully automated induction capabilities. For an analysis of Auto-BN's upper-bound performance when provided with the human-verified ground-truth BNs—which isolates query translation effectiveness from structure induction challenges—please refer to the ablation study in Appendix E.

C.3 NL2SQL (Factual Retrieval)

This baseline tests whether deterministic SQL factual retrieval can support probabilistic inference. Following Gao et al. (2023), we apply a text-to-SQL model that generates a SQL query from the natural language question, using the database schema (same with Gao et al. (2023)) and table description. The result of executing this query (e.g., a count, average, or conditionally filtered subset) is then passed back to the LLM alongside the original question to estimate the desired probability. This tests whether factual retrieval systems can be repurposed for uncertainty estimation, despite lacking probabilistic semantics.

C.4 LLM + Table (Neural-Only)

This baseline probes the LLM's ability to estimate probabilities directly from examples without structured reasoning. Inspired by the BIRD bench-

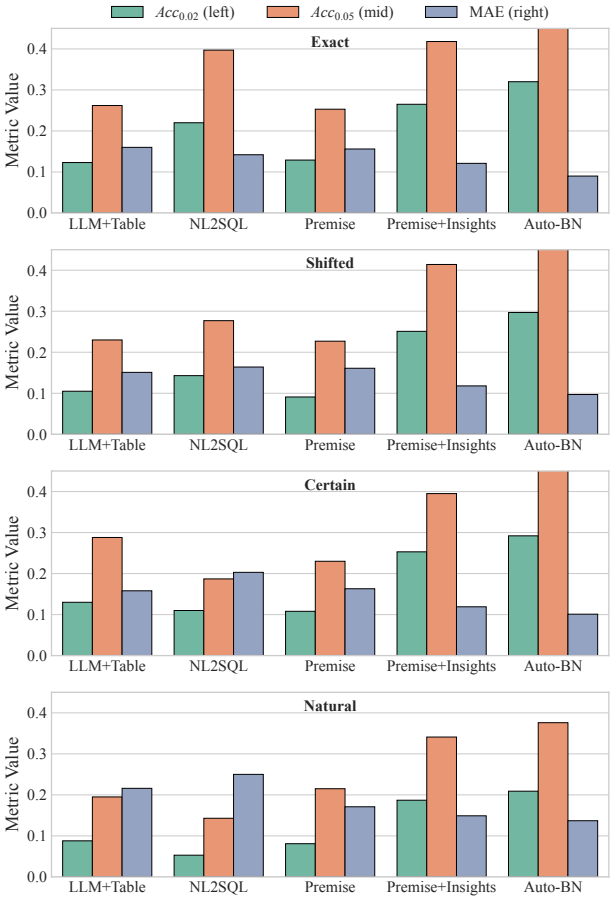


Figure 5: Performance of baselines by question type. Bars represent $Acc_{0.02}$ (↑), $Acc_{0.05}$ (↑), and MAE (↓) for each method.

mark (Li et al., 2023), we concatenate the table schema, table description and table rows with the question, and prompt the LLM for a probability estimate. In the actual situation, due to the large token size of each record, we iteratively append rows to prompt until reach LLM context window limit or 16,384 tokens. This approach avoids any symbolic structure or inference, relying purely on learned statistical patterns in the data.

C.5 TableRAG (Retrieval-Augmented Neural)

To evaluate state-of-the-art retrieval-augmented generation on tabular data, we implement the TableRAG baseline (Chen et al., 2024) using the official Google Research codebase. Unlike the *LLM + Table* baseline, which relies on row sampling to fill the context window, TableRAG leverages query expansion and dense cell retrieval to dynamically extract the most semantically relevant tabular data.

To accommodate the massive scale of the LUCARIO datasets (which average 486k rows and scale up to 4.4M rows), we made two necessary adaptations to the default TableRAG configuration. First, we pre-calculated all cell embeddings offline to construct the CellDB, ensuring efficient dense retrieval during online inference. Second, because probabilistic QA fundamentally requires aggregating broad statistical evidence rather than locating a single factual "needle in a haystack," we increased the `top-k` cell retrieval limit by $10\times$ over the default setting to maximize the evidence provided to the LLM. To ensure strict experimental fairness, TableRAG was evaluated on the exact same hardware configuration as Auto-BN and all other baselines.

C.6 LLM + Premise (Retrieval-Augmented Neural)

To evaluate whether symbolic rules can improve LLM-based reasoning, we follow the CLADDER (Jin et al., 2023) strategy of retrieving premise statements and prompting the LLM to chain them in multiple steps (CoT). Given a question, the LLM first identifies target and evidence variables. We then retrieve up to 40 relevant premises from a pre-generated store, using either BM25, dense vector search, or a hybrid of both (see Appendix B). These premises, alongside the question and table metadata, are passed to the LLM to calculate the answer. This setup tests the model's ability to simulate multi-hop probabilistic derivation from one-hop rules.

Premise retrieval is implemented via LlamaIndex (Liu, 2022) using Sentence-BERT (Reimers and Gurevych, 2019) and BM25. We retrieve up to 40 premises per query.

C.7 LLM + Premise + Insights (Focused Retrieval)

We further enhance the LLM+Premise approach by introducing a focused retrieval mechanism that incorporates only the most informative conditional patterns extracted from the Bayesian network $\mathcal{B}$, termed *insights* (as defined in §3.1). Unlike unfiltered Markov Blanket contexts, which may contain redundant or weak signals compared with the base distribution, this method identifies and prioritizes high-saliency context shifts.

As formally defined in the main text (Eq. 1), the Insight Saliency quantifies the extent to which a context c shifts the belief distribution over a variable X. For each variable, we compute this saliency score across candidate contexts generated from its Markov Blanket (MB). To filter for infor-

Algorithm 1 Insight Extraction from Bayesian Network

Input: Bayesian Network $\mathcal{B}$, thresholds θ
Output: Top-k insights $\mathcal{I}$

Initialize $\mathcal{I} \leftarrow \emptyset$
for each variable $X \in \mathcal{B}$ **do**
  $P_X \leftarrow \text{InferWithoutContext}(\mathcal{B}, X)$
  for $c \in \text{GenerateContexts}(\text{MB}(X), \theta)$ **do**
    $P_{X|c} \leftarrow \text{InferWithContext}(\mathcal{B}, X, c)$
    $s \leftarrow \text{Saliency}(X, c)$ *// Eq. 1*
    valid $\leftarrow \text{IsInsight}(P_{X|c}, P_X, s, \theta)$
    if valid **then**
      Add $(c, P_{X|c}, s)$ to top-k heap H
    end if
  end for
  $\mathcal{I}[X] \leftarrow$ top-k elements from H
end for
return $\mathcal{I}$

mative insights programmatically, we apply a set of criteria via the IsInsight function (Algorithm 1): an insight must (i) exceed a minimum saliency threshold (*significance threshold*, empirically set to 0.5), (ii) avoid flat distributions (*dispersion threshold*, set to 0.08 based on the standard deviation), and (iii) introduce new high-probability modes relative to the base marginal P_X (*new peaks*). The top-k scoring contexts that meet these conditions are retained per node.

To mitigate the combinatorial explosion inherent in large Markov Blankets, we generate conditioning contexts in a staged manner: starting with single-variable contexts and incrementally expanding to include multiple variables. This prioritized enumeration favors simpler contexts first. To ensure tractability, we implement early stopping based on two criteria: (i) a maximum number of context candidates evaluated, and (ii) a target number of valid insights extracted per node. This strategy enables efficient operation on complex, real-world Bayesian networks without exhaustive search.

By restricting the LLM input to this curated set of salient insights rather than an exhaustive collection of premises, the model is able to focus on high-precision, distributionally impactful evidence. This design improves interpretability and generalization, especially in multi-hop probabilistic reasoning scenarios. The full procedure is detailed in Algorithm 1.

C.8 ProbLog-Based Symbolic Baseline

Following QUITE (Schrader et al., 2024), we attempted to encode retrieved premises into ProbLog (De Raedt et al., 2007) programs and evaluate queries symbolically. Conceptually, this baseline is similar to Auto-BN in that both aim to compute calibrated probabilities using a structured probabilistic engine, rather than relying on LLMs to approximate probabilistic reasoning. While theoretically appealing, this method relies on the LLM to perfectly translate each premise needed into ProbLog queries. In QUITE, this led to a failure rate exceeding 90% even under simplified WEP settings with only 40 premises. Due to the high brittleness and low success rate of this approach in LUCARIO, we exclude it from final comparisons.

C.9 Model Details

Three types of LLM are used in our experiments: GPT-4o (Hurst et al., 2024) (unknown size), Llama-3.3-70B (Grattafiori et al., 2024; Meta, 2024) (70 billion parameters), and Mixtral-8x22B (Mistral AI, 2024) (141 billion total and 39 billion active parameters), a sparse mixture-of-experts model (Jiang et al., 2024), via API. During benchmark curation, GPT-4o is used to generate natural language text. All baselines involving LLMs use consistent prompting templates, with GPT-4o serving as the primary model during development. LLaMA-3.3-70B and Mixtral-8x22B are evaluated using the same prompts without additional fine-tuning.

The Bayesian Networks training for Auto-BN uses PGMPY(Ankan and Textor, 2024) library on an r5.4xlarge AWS instance (8 vCPUs, 128 GB RAM, Amazon Linux 2023 system); training time is reported in Appendix D.5

D Additional Experimental Results

D.1 Performance by Question Type

Figure 5 presents the performance of various baselines across different question types. For each type, we report three metrics: $\text{Acc}_{0.02}$ (↑), $\text{Acc}_{0.05}$ (↑), and MAE (↓), capturing both accuracy and error magnitude.

Across all question types, Auto-BN consistently outperforms the other baselines, achieving the highest accuracy and lowest error. Even in the more challenging *Natural* questions, Auto-BN maintains superior performance, highlighting its robustness under linguistic variability. The four variants stress different pipeline steps. *Natural* questions replace

numeric conditions with semantic descriptors (e.g., "high temperature"), which are ambiguous when mapped to table values and BN bin boundaries. *Certain* questions state a single value that must be mapped to its containing BN state (e.g., "exactly 7 goals" $\rightarrow$ bin $(4, 10]$); few or no records match such a point conjunction exactly, so record-counting methods such as NL2SQL break down (Figure 7). *Shifted* questions use an alternative valid range (e.g., "between 4 and 7" when the BN state is $(4, 10]$), which the model must recognize as a sub-range of an existing state. LLM mathematical reasoning has been shown to degrade when numerical values move outside typical ranges (Shrestha et al., 2025), and the narrower range again reduces the number of matching records.

The LLM+Table baseline performs relatively well on the *Certain* question type. This is likely because questions specifying a single numeric value can be more directly matched against sampled rows, allowing the LLM to approximate probabilities more reliably than with binned numeric states.

In contrast, the NL2SQL baseline degrades as the question type becomes more natural or paraphrased, since such formulations are harder to translate into valid and expressive SQL clauses. This confirms the brittleness of deterministic symbolic pipelines under linguistic variation.

Adding Insights consistently improves the LLM+Premise baselines across all question types. This highlights the importance of retrieving high-impact, distributional evidence rather than relying solely on shallow one-hop premises, as it leads to better reasoning fidelity and calibration.

D.2 Performance by Probabilistic Query Type

Table 10 reports zero-shot GPT-4o performance ($\text{Acc}_{0.02}$ and RMSE) across the five probabilistic reasoning types, averaged over all linguistic question variants.

Overall, Auto-BN achieves the strongest performance in 8 of 10 metrics, demonstrating exceptional robustness across *Forward Conditional*, *Diagnostic*, and *Mixed* queries. This proves that exact mathematical inference over an induced graph consistently outperforms purely generative probability estimation. Auto-BN is only outperformed in *Interdependent Evidence* ($\text{Acc}_{0.02}$) and *Mixed+Redundant* (RMSE) by insight-augmented baselines. This stems from an inherent information asymmetry: *Insights* are derived directly from the human-verified Gold BNs. In highly complex or redundant contexts, these oracle-calibrated signals help the LLM filter noise more effectively than Auto-BN's fully automated, purely observational graph.

Among retrieval-based methods, *Insights* consistently outperform raw premises by distilling massive premise spaces into compact, high-impact conditional shifts. Conversely, factual (NL2SQL, LLM+Table) and premise-only baselines fail uniformly on composite queries (*Mixed* types). Such queries require chaining heterogeneous statistical cues and ignoring distractors—capabilities that deterministic SQL or shallow text retrieval fundamentally lack.

D.3 Per-Dataset Results

Table 12 reports MAE by dataset for the main baselines. Auto-BN achieves the lowest or tied-lowest MAE on 8 of 10 datasets. The two exceptions, *coinmarketcap* and *college completion*, favor LLM+Premise(H)+Ins (0.100 vs. 0.117; 0.142 vs. 0.184). These datasets may be harder for automatic structure learning or may benefit more from the Gold-BN-derived premises and insights that Auto-BN does not use. Auto-BN's MAE also varies more across datasets (0.038–0.184) than that of the premise- and insight-based baselines. Adding insights to retrieved premises improves MAE on all 10 datasets.

D.4 Extending the One-Hop Premise Definition

While most prior benchmarks operate on one-hop premises (grounded conditional probability statements of the form $P(Y|X)$ representing direct parent–child dependencies in the underlying conditional graph), we examine whether incorporating multi-hop premises (specifically two-hops) can improve performance by capturing richer contextual dependencies. However, extending to multi-hop contexts increases combinatorial complexity and may introduce noisy or redundant information. To focus the analysis, we restrict our study to two-hop premises and evaluate their impact on the *cookbook nutrition* dataset. We construct two-hop premise representations by extracting all valid, unique two-step conditional dependencies and combining them with existing one-hop premises to form an augmented premise vector store. Both the LLM+Premise(Vector) and its Insight-augmented variant are then evaluated under this expanded evidence setup.

Type	Method	Forward Conditional		Diagnostic		Interdependent Evidence		Mixed		Mixed+ Redundant	
		$Acc_{0.02}$	RMSE	$Acc_{0.02}$	RMSE	$Acc_{0.02}$	RMSE	$Acc_{0.02}$	RMSE	$Acc_{0.02}$	RMSE
Factual	LLM + Table	9.6	0.234	12.8	0.212	10.2	0.229	13.9	0.248	11.5	0.230
	TableRAG	6.6	0.304	5.5	0.297	10.2	0.258	8.4	0.319	5.5	0.320
	Reasoning (GPT-5.4)	11.0	0.268	12.1	0.276	12.2	0.281	11.4	0.304	10.0	0.287
	NL2SQL	12.6	0.238	16.6	0.242	13.1	0.261	16.4	0.285	8.9	0.249
Prob. Premises	LLM+Premise(Vector)	10.3	0.207	8.8	0.237	7.8	0.277	8.0	0.255	6.1	0.228
	LLM+Premise(BM25)	9.0	0.218	9.0	0.261	5.7	0.259	10.5	0.277	10.0	0.231
	LLM+Premise(Hybrid)	11.0	0.195	10.7	0.235	7.6	0.241	11.7	0.252	10.6	0.198
Prob. Insights	LLM+Premise(Vector)+Insights	17.2	0.179	19.8	0.211	**39.0**	0.202	19.5	0.218	24.8	0.177
	LLM+Premise(BM25)+Insights	13.9	0.193	19.3	0.217	38.6	0.182	17.6	0.239	25.0	0.186
	LLM+Premise(Hybrid)+Insights	15.0	0.182	21.3	0.218	37.2	0.181	20.3	0.232	24.1	**0.167**
	Auto-BN	**28.7**	**0.151**	**31.1**	**0.151**	32.9	**0.152**	**22.3**	**0.208**	**25.1**	0.177

Table 10: Performance across probabilistic query types using GPT-4o. Each method is grouped by baseline type and evaluated on $Acc_{0.02}$ (↑) and RMSE (↓) per reasoning category.

Method	Premises	% $Acc_{0.02}$ ↑	% $Acc_{0.05}$ ↑	MAE ↓	RMSE ↓	% Error Rate ↓
LLM+Premise(Vector)	1-hop	8.2	24.4	0.162	0.226	0.0
	2-hop	9.2	24.1	0.165	0.231	0.0
	Improvement	**0.9**	**-0.3**	**0.003**	**0.005**	**0.0**
LLM+Premise(Vector)+Insights	1-hop	25.7	42.7	0.104	0.173	0.0
	2-hop	24.4	40.2	0.113	0.184	0.0
	Improvement	**-1.3**	**-2.6**	**0.008**	**0.011**	**0.0**

Table 11: 1-Hop vs 2-Hop Premises Ablation Study on *cookbook nutrition* Dataset

Table 11 summarizes the ablation results. For the LLM+Premise(Vector) variant, introducing 2-hop premises yields a minor gain in $Acc_{0.02}$ (+0.9), but slightly degrades $Acc_{0.05}$, MAE, and RMSE, indicating potential trade-offs between local accuracy and global calibration.

Interestingly, the Insight-augmented method performs better when restricted to 1-hop premises, with performance declining across all metrics when 2-hop premises are included in this dataset. This suggests that the insight extraction mechanism, designed to identify salient signals, may be adversely affected by the increased noise and reduced contextual relevance of deeper dependency chains.

These preliminary results suggest that, although multi-hop premises can provide longer reasoning chains and broader contextual coverage, their utility appears to depend strongly on precision and relevance. In the absence of strong filtering mechanisms, deeper contexts may introduce noise that reduces the quality of retrieved evidence and hinders downstream reasoning. Further evaluation across diverse datasets is needed to determine when and how multi-hop contexts can be incorporated effectively.

D.5 Running Time Analysis

Auto-BN comprises an offline training phase (dependency graph discovery and BN parameter learning) and an online inference phase (robust query translation and exact probabilistic execution). Table 13 summarizes the computational footprint of both phases across the benchmark.

Offline Induction Cost. As shown in Table 13, the one-time offline induction scales naturally with dataset complexity. It ranges from 94 seconds (*movies*) to 5,950 seconds (*coinmarketcap*), the latter requiring extensive structural search over numerous attributes. Crucially, this induction is a CPU-only process executed exactly once per table. When amortized over the lifespan of a deployed database, this offline cost becomes negligible.

Online Inference Latency. During the online phase, inference latency averages a highly efficient 1.90 seconds per query across the benchmark. To maintain this low overhead despite the rigorous Multi-Path Cross-Consistency Voting (§4.2), Auto-BN executes the $N{=}3$ LLM parsing calls concurrently, bounding wall-clock time strictly by the slowest individual API response. The only notable

Dataset	LLM+Table	Reasoning	NL2SQL	LLM+Premise(H)	+Insights	Auto-BN
airlines	0.175	0.212	0.200	0.140	0.084	**0.038**
bike stations weather	0.154	0.218	0.150	0.165	0.148	**0.099**
coinmarketcap	0.235	0.245	0.237	0.133	**0.100**	0.117
college completion	0.164	0.201	0.198	0.172	**0.142**	0.184
cookbook nutrition	0.167	0.170	0.227	0.156	0.104	**0.102**
hockey scoring	0.188	0.190	0.252	0.214	0.171	**0.120**
movies	0.151	0.164	0.145	0.131	0.116	**0.090**
basketball players	0.169	0.226	0.161	0.213	0.171	**0.123**
sales in weather	0.164	0.209	0.167	0.166	0.144	**0.101**
twitter	0.145	0.152	0.161	0.139	**0.090**	**0.090**

Table 12: Per-dataset MAE (↓). All methods use GPT-4o except Reasoning (GPT-5.4). LLM+Premise(H) uses Hybrid retrieval; +Insights adds insights to it. Best per row in bold.

Dataset	Training (s)	Inference (s)	Total (s)	Inference/Q (s)	Total/Q (s)
airlines	242	935	1,177	1.49	1.87
basketball players	852	1,022	1,874	1.63	2.98
bike stations weather	1,057	1,084	2,141	1.73	3.41
coinmarketcap	5,950	1,067	7,017	1.70	11.17
college completion	218	981	1,199	1.56	1.91
cookbook nutrition	287	1,038	1,325	1.65	2.11
hockey scoring	632	1,171	1,803	1.87	2.87
movies	94	975	1,069	1.55	1.70
sales in weather	1,176	2,569	3,745	4.09	5.96
twitter	100	1,065	1,165	1.70	1.86
Mean	1,061	1,191	2,252	1.90	3.58

Table 13: Training and inference time statistics for Auto-BN pipeline using GPT-4o across datasets (157 questions $\times$ 4 question types = 628 queries per dataset). Training is a one-time offline cost. Inference/Q and Total/Q denote mean seconds per query.

outlier is *sales in weather* (4.09 s/q), where exceptionally high BN edge density inherently increases the computational complexity of exact variable elimination. Overall, this 1.90-second latency is highly competitive with LLM+RAG methods (e.g. TableRAG $\sim$4.51 s/q) while delivering the critical advantage of mathematically grounded, hallucination-free probabilistic guarantees.

D.6 Ablation Study: Robust Query Translation

A critical bottleneck in neuro-symbolic pipelines is the inherent brittleness of LLM-based semantic parsing. To demonstrate how Auto-BN overcomes this limitation, we ablate our robust query translation and multi-level relaxation mechanisms (§4.2, §4.3) and analyze the resulting failure modes. We define a **Base Auto-BN** configuration that relies exclusively on a single zero-shot parse, completely removing multi-path voting, execution-guided refinement, and all inference-level fallbacks. Figure 6 illustrates the exact error distribution for this uncorrected baseline across all benchmark datasets.

The dominant failure categories are: *State Invalid for Node* (62.3%), where the LLM predicts a state that does not belong to the queried node; *State Not Found* (28.3%), where the predicted state does not exist anywhere in the BN vocabulary; *Variable Conflict* (3.0%), where the same variable appears as both target and evidence; *Node Not in Graph* (2.2%); *Node Name as State* (1.1%); *JSON Parsing errors* (1.1%); and *Other Inference Errors* (1.9%).

Over 90% of these baseline failures stem from node–state vocabulary mismatches, confirming that without structural safeguards, LLMs systematically hallucinate states outside the BN schema. These findings directly motivate our proposed pipeline: (i) multi-path cross-consistency voting to surface schema-invalid parses early, (ii) execution-guided iterative refinement to self-correct mismatches, and (iii) semantic relaxation fallbacks to recover the nearest valid node or state when strict parsing fails.

Table 14 quantifies the impact of these mechanisms across all three LLMs on the full LUCARIO benchmark, with an *Improvement* row showing the absolute delta (Auto-BN − Base Auto-BN) per LLM. By successfully insulating the exact probabilistic inference engine from the stochastic brittleness of LLM generation, the full Auto-BN framework reduces the error rate from as high as 20.8%

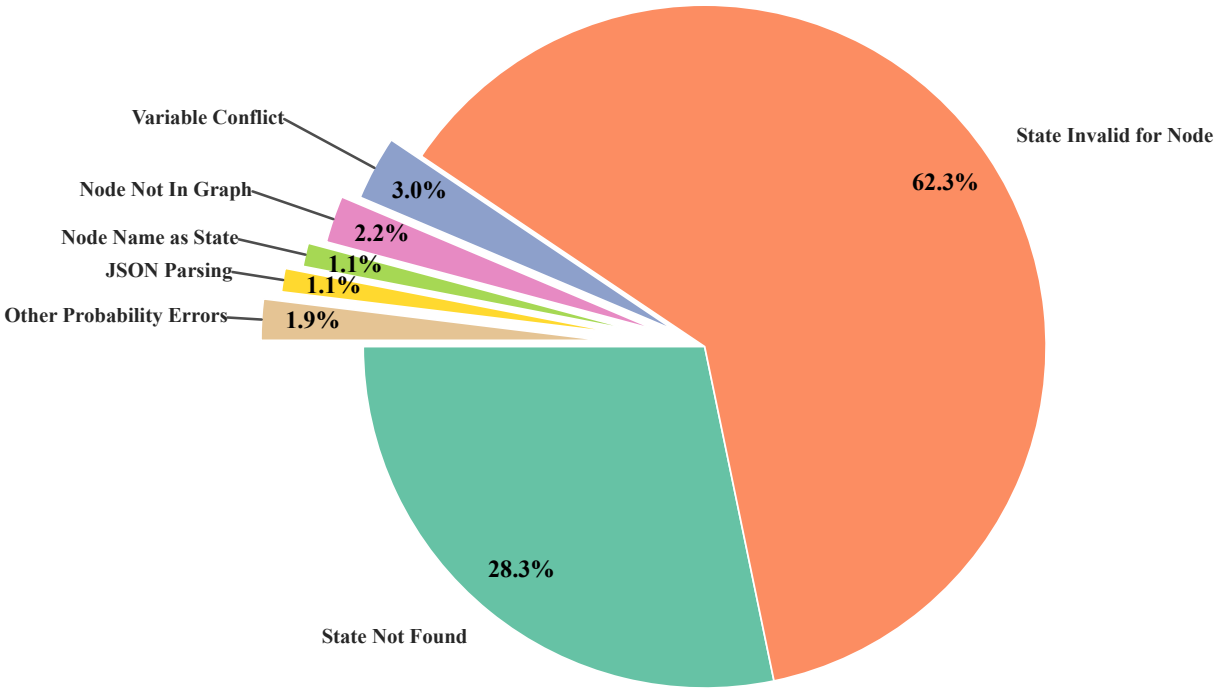


Figure 6: Distribution of parsing errors in the Base Auto-BN ablation configuration (single-pass parse, no correction mechanisms). Percentages are averaged across all datasets. Auto-BN resolved these errors.

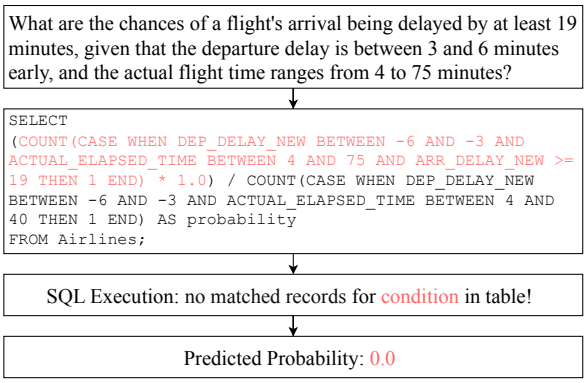


Figure 7: Case Study for Factual Method on Prob. QA

down to near-zero, while simultaneously drives consistent gains in strict accuracy ($Acc_{0.02}$ and $Acc_{0.05}$) and maintaining stable MAE and RMSE.

D.7 Case Study

Why factual retrieval based method doesn't work well on prob table QA? Figure 7 shows one frequent failure that NL2SQL baseline made. Even if NL2SQL model successfully translates the probabilistic question into valid SQL query, the context provided in the question might not find exact match records in the table. That causes NL2SQL baseline frequently output 0.0 (no records) or 1.0 (bias in retrieved few records).

E Gold BN vs Induced BN

In our main evaluation, the proposed Auto-BN framework operates in a strict, fully automated setting—inducing its Bayesian Network directly from raw tabular data without human intervention. To establish the upper bound of this benchmark and isolate the impact of automatic structure induction, we compare the standard Auto-BN against an ablation using the human-verified Gold BN.

This analysis serves two purposes: First (§ E.1), it establishes one type of benchmark's performance ceiling when the underlying mathematical structure is perfectly specified. Second (§ E.2), it quantifies the structural differences between the induced and Gold BNs, confirming that Auto-BN operates without information leakage and validating its strong performance in the harder, fully automated regime.

E.1 QA Performance: Benchmark Upper Bound

To establish the upper bound, we introduce **Auto-BN (Gold)**. This ablation uses the exact same execution-guided translation, voting, and multi-level fallback pipeline described in § 4, but replaces the automatically induced BN with the benchmark's human-verified ground-truth BN.

When equipped with the human-verified BN, Auto-BN (Gold) with GPT-4o achieves an out-

LLM	Method	% $\text{Acc}_{0.02}$ ↑	% $\text{Acc}_{0.05}$ ↑	MAE ↓	RMSE ↓	% Err ↓
GPT-4o	Base Auto-BN	26.4 ± 14.6	45.3 ± 15.3	0.107 ± 0.040	0.168 ± 0.055	5.6 ± 4.6
	Auto-BN	28.0 ± 15.5	46.7 ± 16.2	0.107 ± 0.043	0.169 ± 0.058	0.0 ± 0.0
	Improvement	$+1.6$	$+1.4$	0.000	$+0.001$	-5.6
Llama-3.3-70B	Base Auto-BN	23.6 ± 12.8	41.6 ± 14.4	0.111 ± 0.036	0.169 ± 0.047	14.8 ± 10.6
	Auto-BN	25.4 ± 13.6	44.8 ± 15.0	0.109 ± 0.040	0.168 ± 0.053	0.0 ± 0.0
	Improvement	$+1.8$	$+3.2$	-0.002	-0.001	-14.8
Mixtral-8x22B	Base Auto-BN	22.7 ± 13.9	40.4 ± 15.1	0.112 ± 0.033	0.167 ± 0.043	20.8 ± 12.0
	Auto-BN	26.1 ± 15.4	44.6 ± 16.1	0.112 ± 0.046	0.174 ± 0.062	0.0 ± 0.2
	Improvement	$+3.4$	$+4.2$	0.000	$+0.007$	-20.8

Table 14: Ablation of robust query translation on LUCARIO. Auto-BN (this work) adds multi-path voting, execution-guided refinement, and inference-level fallbacks over Base Auto-BN (single-pass parse, no correction). *Improvement* rows show absolute deltas (Auto-BN − Base Auto-BN); negative values for MAE/RMSE/Err indicate improvement.

Method	LLM	% $\text{Acc}_{0.02}$ ↑	% $\text{Acc}_{0.05}$ ↑	MAE ↓	RMSE ↓
Auto-BN (Gold)	GPT-4o	**77.9**	**83.8**	**0.034**	**0.085**
	Llama-3.3-70B	66.4	75.4	0.049	0.108
	Mixtral-8x22B	67.5	76.4	0.050	0.111
Auto-BN (Induced)	GPT-4o	28.0	46.7	0.107	0.169
	Llama-3.3-70B	25.4	44.8	0.109	0.168
	Mixtral-8x22B	26.3	45.4	0.109	0.168

Table 15: QA performance ablation isolating the impact of graph structure quality. *Auto-BN (Gold)* represents the upper bound when provided an expert-curated schema. Metrics are averaged over 6,280 total questions.

standing $\text{Acc}_{0.05}$ of 83.8% and a near-zero MAE of 0.034. This establishes a strong upper bound for the benchmark, demonstrating that the proposed query translation pipeline is highly effective when the underlying probabilistic structure is accurately specified.

By contrast, the standard Auto-BN (Induced) operates at an MAE of roughly 0.107. Since the translation prompts and fallback mechanisms are strictly identical in both settings, this performance gap reflects the inherent difficulty of purely observational structure induction. However, achieving an MAE of 0.107 in a purely automated, zero-shot setting remains a highly competitive result, demonstrating that Auto-BN provides grounded, reasonable probabilistic estimates even without expert domain knowledge.

E.2 Structural Fidelity of Induced BNs

To structurally explain the performance gap observed above, we evaluate how well the data-driven PC algorithm recovers the Gold graph structure. We compare the edge sets of the Induced BN against the Gold BN across all 10 datasets, computing precision (P), recall (R), and F_1 scores in two settings: (1) **Directed**: an induced edge $u \to v$ is a true positive only if the gold graph contains the exact same directed dependency; and (2) **Undirected**: an induced edge between u and v is a true positive if the gold graph connects them in either direction.

	Directed			Undirected		
	P	R	F_1	P	R	F_1
Mean	0.165	0.156	0.159	0.582	0.563	0.569
Std	0.105	0.105	0.103	0.236	0.266	0.248

Table 16: Edge-level structural fidelity of the automatically induced BN vs. the human-verified Gold BN.

Structural Divergence and Robustness. The undirected F_1 score averages 0.569, indicating that the automatic induction algorithm successfully identifies over half of the underlying statistical associations between variables. This is a robust recovery rate, particularly considering that there are often multiple valid ways to represent similar hidden relationships among columns (e.g., modeling a direct 1-hop dependency via an intermediate 2-hop path). However, the directed F_1 drops severely to 0.159. This profound structural divergence mathematically confirms that the automatically induced BN is distinctly different from the human-verified Gold BN, definitively proving there is no leakage of expert domain knowledge into our automated pipeline. It also validates that Auto-BN and premise-based baselines (e.g., LLM+Premise)

operate under fundamentally different information conditions.

While Auto-BN can recover from semantic vocabulary mismatches, it cannot mathematically compensate for systematically inverted dependency directions in the induced graph. In light of this significant structural handicap, Auto-BN's ability to still achieve an MAE of 0.107 underscores the exceptional robustness of the proposed framework, proving its practical viability and strong performance for purely data-driven tabular reasoning.

F Auto-BN-Tool (Agentic BN Inference)

To facilitate the integration of our proposed system into modern agentic workflows, we introduce **Auto-BN-Tool**, a tool-calling extension of the base Auto-BN framework. By wrapping the exact probabilistic inference engine into an executable tool, Auto-BN-Tool allows the LLM to operate within a multi-turn agent loop. The model can iteratively invoke the tool, analyze execution results or error messages, and synthesize a final answer from multiple sub-queries. This formulation demonstrates the flexibility of our modular design, showing that Auto-BN can seamlessly serve as a foundational probabilistic reasoning engine for autonomous LLM agents.

In practice, at each turn, the LLM is prompted with the natural language question alongside the derived BN metadata (i.e., node and state vocabularies) and the history of prior tool executions. The LLM is explicitly instructed to decompose complex questions into manageable sub-queries. It then either emits a `<tool_call>` block to compute a specific probability on the BN or outputs the final calculated answer. The agentic loop terminates when the LLM produces a final answer or reaches a predefined budget ($T_{\max}$=5 turns, with a maximum of $C_{\max}$=3 tool calls per run).

This iterative formulation offers two primary advantages. First, it enables the LLM to handle complex, multi-step reasoning by issuing sequential sub-queries (e.g., verifying a marginal distribution before querying a conditional one). Second, it acts as an intrinsic error-correction mechanism. If the LLM hallucinates an invalid node or state, the tool returns an execution error. The agent can then observe this feedback, consult the provided BN vocabulary, and self-correct its semantic mapping in the subsequent turn.

We evaluate Auto-BN-Tool using GPT-4o as the backbone LLM. As detailed in Table 17, the agentic framework delivers strong overall performance, achieving an $Acc_{0.05}$ of 47.6%, a MAE of 0.105, and an RMSE of 0.166. Furthermore, its strict accuracy (27.8% $Acc_{0.02}$) remains highly competitive with the base Auto-BN pipeline (28.0%) while successfully maintaining a near-zero error rate (0.1%). Ultimately, these results demonstrate that integrating Auto-BN into an autonomous agent loop not only preserves the rigorous, hallucination-free grounding of exact inference, but also maintains system robustness and calibration through multi-turn self-correction and compositional query decomposition.

G Prompt Templates

This appendix provides the exact prompt templates used in our experiments. Runtime variables are denoted in lowercase within curly braces, e.g., `{question}`.

G.1 Auto-BN: Semantic Parsing Prompt

The following system prompt is used to translate a natural language question into a structured probabilistic query (target node, target state, evidence context) for exact BN inference. This prompt is used by **Auto-BN**.

Auto-BN Semantic Parsing Prompt

You are an assistant that maps natural language questions into probabilistic components of a Bayesian Network. The Bayesian Network is built from a tabular data, where each node corresponds to a column in the original table.

A Bayesian Network consists of:
- **Nodes** (random variables)
- **States** (possible values each node can take)
- **Directed edges** indicating conditional dependencies between variables

This network encodes a joint probability distribution P(X_1, X_2, ..., X_n). Each edge A → B means that the conditional probability P(B | A) is directly represented.

Nodes and their states: {nodes}
Edges (parent → child): {edges}
Descriptions of the nodes (from the tabular data): {table_description}

Given the following question, extract:
(1) The **target node** being queried (the variable we want the probability for)
(2) The **target state** or value of that variable
(3) Any **observed context** or evidence described in the question (as a dictionary of node → state)

Question: {question}

CRITICAL INSTRUCTIONS:
- You MUST use ONLY the exact node names and

Method	% $\text{Acc}_{0.02}$ ↑	% $\text{Acc}_{0.05}$ ↑	MAE ↓	RMSE ↓	% Error ↓
Auto-BN	$\mathbf{28.0 \pm 15.5}$	46.7 ± 16.2	0.107 ± 0.043	0.169 ± 0.058	$\mathbf{0.0 \pm 0.0}$
Auto-BN-tool	27.8 ± 15.7	$\mathbf{47.6 \pm 16.5}$	$\mathbf{0.105 \pm 0.043}$	$\mathbf{0.166 \pm 0.057}$	0.1 ± 0.4

Table 17: Performance of the agentic Auto-BN-Tool framework on the Lucario benchmark.

```
state values provided above
- Do NOT create new state values or modify existing ones
- If the question mentions a range that does not exactly match a provided state, choose the CLOSEST available state
- Double-check that your selected states exist in the provided list before responding
Respond ONLY with a JSON object in the exact format below:
{ "target_node": "node_name", "target_state": "state_value", "context": {"node1": "state1"} }
```

G.2 Auto-BN-Tool: Agentic Loop Prompt

The following system prompt governs the Auto-BN-Tool agentic loop (Appendix F). The LLM reasons step-by-step, issuing `<tool_call>` blocks to invoke exact BN inference, observing results, and emitting a final answer via `<final_answer>` tags.

Auto-BN-Tool Agentic System Prompt

```
You are a probabilistic reasoning agent that answers questions about tabular data using a pre-learned Bayesian Network (BN).

=== Bayesian Network Structure ===
Nodes and their possible states: {nodes}
Edges (parent → child): {edges}
Column descriptions: {table_description}

=== Tool: query_bayesian_network ===
To invoke the tool, output a JSON block wrapped in <tool_call> tags:
<tool_call>{"target_node": "NodeName", "target_state": "StateValue", "context": {"N1": "S1"}}</tool_call>
You may make up to {max_tool_calls} tool call(s). The tool returns:
{"status": "success"/"error", "probability": 0–1 or null, "fallback_used": ...}

=== Providing your final answer ===
Mark your final answer with:
<final_answer>0.42</final_answer>
The value must be a number in [0, 1]. Nothing should appear after this tag.

=== Reasoning strategy ===
(1) Identify the target variable, its value, and any conditioning evidence
(2) Decompose into sub-queries only when the question explicitly requires combining independent probabilities (e.g., computing P(A) and P(B|C) separately), or when an initial call fails and reformulation is needed; simple conditional queries P(A|B) should be issued as a single tool call
(3) Call query_bayesian_network with the identified target and context
(4) If the tool returns an error, check node/state names against the schema and retry with corrected names
(5) Synthesize the sub-query results into a final probability estimate and emit your final answer using <final_answer>X</final_answer>
```