CausalGraphBench: a Benchmark for Evaluating Language Models capabilities of Causal Graph discovery

Nikolay Babakov
CiTIUS, Universidade de Santiago de Compostela
nikolay.babakov@usc.es

Ehud Reiter
University of Aberdeen
e.reiter@abdn.ac.uk

Alberto Bugarín
CiTIUS, Universidade de Santiago de Compostela
alberto.bugarin.diz@usc.es

Abstract

This paper introduces CausalGraphBench, a benchmark designed to evaluate the ability of large language models (LLMs) to construct Causal Graphs (CGs), a critical component of reasoning models like Bayesian Networks. The benchmark comprises 35 CGs sourced from publicly available repositories and academic papers, each enriched with detailed metadata to facilitate systematic and consistent evaluation. We explore various LLM-driven methods for CG discovery, analyzing their performance across different graph sizes and complexity levels. Additionally, we examine the effects of data contamination on the quality of the generated CGs.

Our findings reveal that methods relying on approaches with a limited number of queries to LLM, particularly those leveraging the full graph context, consistently outperform query-intensive and exhaustive approaches, which tend to overemphasize local relationships. Across all methods, performance declines as graph size increases.

1 Introduction

Recent advances in large language models (LLMs) have expanded their applications into domains not traditionally associated with natural language processing (e.g. education (Kasneci et al., 2023), programming (Guo et al., 2024)). One such domain is using LLMs to build Causal Graphs (CG), essential for causal models like Bayesian networks (BNs) (Koller, 2009). A growing body of research demonstrates that LLMs can effectively address various CG-related tasks (Wang et al., 2024; Chen et al., 2024) and can even construct these graphs (Wan et al., 2024), a task often referred to as Causal Graph discovery (CGD). Traditionally, this task has been tackled using structure learning algorithms (Kitson et al., 2023), which derive the graph from data, or through expert elicitation (Nyberg et al., 2022), where human expertise guides the construction of the CG.

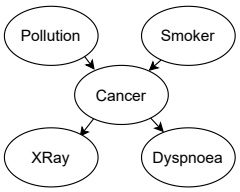

Figure 1: Causal Graph of the BN related to the lung cancer problem (Korb and Nicholson, 2010).

CGs are typically represented as directed acyclic graphs (DAGs), illustrating variables and their causal dependencies. For instance, consider the example shown in Figure 1, which depicts a CG of a simple BN (Korb and Nicholson, 2010). This BN models a hypothetical scenario involving potential causes (e.g., Pollution and Smoker) and effects (e.g., X-Ray results and Dyspnoea) of Lung Cancer.

Our work focuses on methods that utilize LLMs for the CGD task, which infer causal links based purely on CG node names. Several related approaches have been proposed (Ban et al., 2023b; Jiralerspong et al., 2024; Babakov et al., 2024; Cohrs et al., 2024; Zhang et al., 2024). Still, their evaluations often lack consistency, as different studies employ distinct sets of CGs, making direct comparisons challenging (see Appendix Table 7 for details on the CGs used in these studies).

To address this limitation, we introduce CausalGraphBench, a unified benchmark designed to evaluate and compare the capabilities of LLMs in CGD[1]. The benchmark consists of 35 CGs from the literature. We use this benchmark to evaluate the performance of currently proposed LLM-driven CGD methods, providing a comprehensive

[1] https://gitlab.nl4xai.eu/nikolay.babakov/causal-graph-bench

comparison across approaches. Additionally, as an auxiliary validation task, we perform a detailed assessment of data contamination to ensure the robustness and reliability of the results.

2 Related works

LLMs have been explored for solving graph-related tasks such as connectivity, cycle detection, shortest path, topological ordering, and other graph problems (Wang et al., 2024; Chen et al., 2024).

LLMs have also been applied to construct CGs. One approach involves first using data-driven methods to build an initial structure and then refining it with LLMs. For example, the ILS-CSL framework (Ban et al., 2023a) iteratively refines data-driven CGs by using LLMs to validate and correct causal relationships, incorporating edge-specific constraints for improved accuracy. Similarly, a method proposed in (Long et al., 2023a) uses LLMs as "imperfect experts" to orient ambiguous edges within a Markov equivalence class, leveraging a Bayesian framework to ensure consistency and manage risks.

Another branch of research uses LLMs to construct causal graphs directly, following either exhaustive querying or minimal-query approaches. Exhaustive methods query all possible node pairs or triplets, as seen in (Cohrs et al., 2024), which employs LLMs as conditional independence oracles, and (Zhang et al., 2024), which integrates Retrieval-Augmented Generation and majority voting. Vashishtha et al. (2023) extends this by merging triplet-based subgraphs, while other works explore similar pairwise querying strategies (Long et al., 2023b; Kıcıman et al., 2023; Feng et al., 2024; Darvariu et al., 2024; Zhou et al., 2024). In contrast, minimal-query approaches aim to construct the full graph with fewer interactions. Jiralerspong et al. (2024) iteratively builds the structure starting from root nodes, Ban et al. (2023b) follows a structured three-step process including self-evaluation, and Babakov et al. (2024) introduces LLM-experts that independently generate graphs, with final structures determined by majority voting.

To the best of our knowledge, there has been only one attempt to establish a benchmark for evaluation of LLM-driven CGD, proposed by Zhou et al. (2024). This benchmark was limited to publicly available CGs and did not include a comparative evaluation of existing LLM-based methods. Furthermore, the fact that all CGs are easily accessible in scrapable form on websites like bnlearn.com raises concerns about potential data contamination, which could compromise the validity of the results.

3 Benchmark information

3.1 Task statement

In this section, we formally define the task of Causal Graph discovery, which the collected benchmark is designed to evaluate. Let $G = (\mathcal{V}, \mathcal{E})$ denote a DAG, where $\mathcal{V}$ is the set of nodes (or variables) and $\mathcal{E}$ is the set of directed edges. Each node $v_i \in \mathcal{V}$ corresponds to a named variable, and each directed edge $e_{i,j} \in \mathcal{E}$ represents a causal effect from node v_i to node v_j. The goal of Causal Graph discovery is, given only the names of the nodes $\mathcal{V}$, to determine the set of edges $\mathcal{E}$ that form the DAG G. Formally, this can be expressed as constructing a graph $G^* = (\mathcal{V}, \mathcal{E}^*)$, where $\mathcal{E}^*$ is the set of causal relationships between the nodes extracted solely relying on the semantic of the variable names.

3.2 Data collection

In our work, all CGs are parts of Bayesian Networks. The information about the BNs was collected from two main sources. The first source was the well-known bnlearn[2] repository, which hosts extensive collections of BNs. The second source comprised academic papers on constructing specific BNs for particular tasks studied in the BN-related survey by Babakov et al. (2025). This initially resulted in 163 CGs. The main criteria for including certain CG into the benchmark was the feasibility of obtaining the correct structure of CG and its metadata. The main obstacle was the absence of the runnable file associated with the papers. We located only 19 CGs with the runnable file (i.e. CG had the files in one of the popular formats, such as bif, net, etc., so we can load it directly on our machine), 14 of which come from bnlearn website. The other CGs were presented visually in the papers, so the extraction of the structure from them was possible only by visual studying of the presented CG scheme. Thus, we considered only medium-sized CGs (as a rule, no more than 20 nodes) to make such extraction possible and less prone to mistakes. The CG selection diagram and the list of all included CGs are shown in Appendix Figure 5 and Table 8 correspondingly.

To make each CG suitable for the task of CGD, we collected comprehensive metadata. The meta-

[2] bnlearn.com

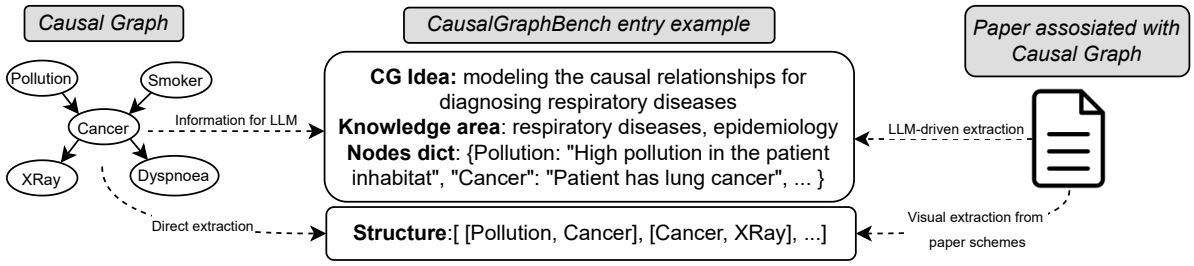

Figure 2: Example of creation of benchmark entry in CausalGraphBench. The node names of CG together with the corresponding paper's content are shown to GPT-4o, which is queried to extract the key metadata describing the CG: the general idea of a CG, knowledge area, and nodes dictionary. The structure of the graph is extracted either from the CG file or if unavailable manually from the paper's content.

data includes the following: *CG idea*, which describes the primary context or problem modelled by the CG (e.g., diagnosing respiratory diseases); *Knowledge Area*, specifying the broader domain the CG belongs to, such as epidemiology or respiratory diseases; *Nodes Description*, a dictionary that maps the node names as they are represented in the original CG to unambiguously defined names; and *Graph Structure*, which lists the directed edges between nodes that define the causal relationships.

Lacking in-depth expertise in the domains of most CGs included in the benchmark, we relied on an LLM (OpenAI GPT-4o) and available CG information to extract the CG idea, knowledge area, and nodes dictionary. Providing all available information about the necessary CG (names of the nodes and the content of the paper describing this CG), we consecutively prompted LLM to extract each part of the metadata (i.e. one prompt for CG idea, another prompt for knowledge area, and the last one for nodes dictionary). The exact structure of CG was either taken from the CG file associated with the paper or constructed manually according to the scheme of CG. Figure 2 shows the scheme of metadata extraction and the example of the resulting entry. The exact prompts and an example of the extracted metadata are shown in Appendix B and C correspondingly.

3.3 Data statistics

Table 1 presents the statistics of the collected benchmark. Of the 35 CGs included, 14 were obtained from publicly available repositories, while 21 were sourced from academic papers. Publicly available CGs are generally larger, with a median of 42 nodes and 59 edges, compared to 14 nodes and 17 edges for CGs from papers. This difference arises because papers rarely provide runnable CG files (Babakov et al., 2025), often requiring the structure to be manually extracted from graphical representations. Such tasks are typically only feasible for smaller graphs.

	Publicly available	**From papers**	**All**
CGs count	14	21	35
Nodes count, median	42	14	16
Edges count, median	59	17	21

Table 1: Collected benchmark statistics

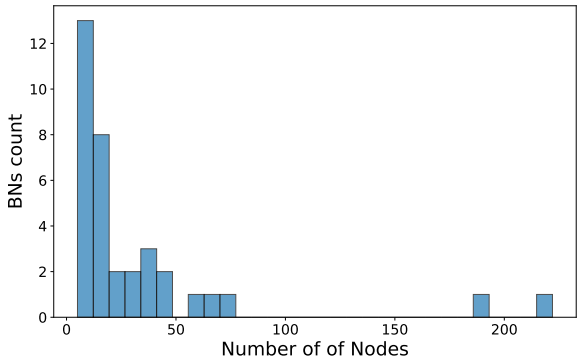

Figure 3: Histogram of the number of nodes in the BNs' Causal Graphs included in the CausalGraphBench.

Figure 3 illustrates the distribution of the number of nodes in the benchmark. Most CGs have fewer than 50 nodes, with only a few outliers exceeding 200 nodes.

4 Experimental setup

4.1 Causal Graph discovery methods

In our experiments, all methods have the same standardised information about each CG, as defined in the benchmark. This includes the *CG idea*, the *knowledge area*, and the *nodes dictionary* - a list of

node names mapped to their clarified form to avoid ambiguity. Each query to the LLM is constructed using this information to ensure fairness across the methods.

The *baseline* method involves a single query to the LLM, asking it to generate a list of edges that form the CG. No further guidance or additional instructions are provided to refine the output. The *harn* method (named by the corresponding paper's title), derived from (Ban et al., 2023b), builds upon the *baseline* approach by incorporating an additional step. After generating the initial structure in the first query, the LLM is asked to evaluate the generated edges and remove those deemed incorrect.

The *pair* method queries the LLM for each possible pair of nodes in the BN, asking whether a causal relationship exists between them. Similarly, the *triplet* method (Vashishtha et al., 2023) extends this approach by querying all possible triplets of nodes. For each triplet, the LLM is expected to generate the subgraph that includes the corresponding nodes or indicate if any nodes are isolated due to a lack of causal relationships. These methods are resource-intensive; for a CG with N nodes, the *pair* method requires $\binom{N}{2} = \frac{N \cdot (N-1)}{2}$ queries, while the *triplet* method requires $\binom{N}{3} = \frac{N \cdot (N-1) \cdot (N-2)}{6}$ queries. Due to this computational cost, we restrict experiments with these methods to CGs containing no more than 10 nodes.

The *efficient* method (Jiralerspong et al., 2024) constructs the Causal Graph by iteratively expanding and inserting causal relationships. The first query extracts the nodes identified as independent. Then, the method prompts the LLM to generate the set of variables causally affected by the current node, gradually building the graph. Each expansion query includes the cumulative graph structure from previous steps, ensuring consistency. For each predicted edge, a cycle-check is performed before adding it to the graph, preserving the directed acyclic nature of a CG. Although this approach is significantly more efficient than the *pair* and *triplet* methods, requiring only $O(N)$ queries, the accumulation of results in successive queries can become computationally demanding. To balance efficiency and feasibility, we apply this method only to CGs with up to 50 nodes.

The *delphi* method (Babakov et al., 2024) leverages multiple "LLM-experts", each tailored to the knowledge area of the CG, to collaboratively construct the Causal Graph. The profiles for these experts are specifically generated to align with the required knowledge domain, ensuring their expertise is relevant to the task. In our setup, we select three experts as a hyperparameter. Each expert is queried with two consecutive prompts: first, to think step-by-step about the causal relationships between all nodes in the CG, and second, to organise the identified relationships into a valid JSON format. The final CG is formed by majority voting, where an edge is included if the majority of experts agree on its existence. Additionally, the method incorporates further queries to check for and prevent cycles in the graph, ensuring it remains a valid DAG.

The *finetune* method involves fine-tuning LLMs specifically for the task of CGD. The prompts are prepared in a manner similar to the *baseline* method, where the input includes essential CG information, and the expected output is a correct CG. For each CG, a separate model is trained using the remaining CGs as training and validation data, with an 80-20% split stratified by the number of nodes. The detailed fine-tuning setup for each LLM will be presented alongside the descriptions of the LLMs engaged in the experiments.

Most methods included in the experiments are taken from the literature search (*harn, triplet, efficient, delphi*). The *pairwise* method could be referred to many papers discussed in Section 2, most of which rely on a similar exhaustive setup. *Finetune* and *baseline* methods are taken just relying on common knowledge of performing the experiments with benchmarks. The examples of the prompts related to all described methods are available in Appendix D.

4.2 Language models

In our experiments, we utilize one proprietary LLM, GPT-4o, and two open-sourced models: Llama-3.3-70B-Instruct (Llama-3.3)[3] and Llama-3.1-8B-Instruct (Llama-3.1)[4].

GPT-4o is fine-tuned using OpenAI's proprietary tuning features. For Llama-3.1, we apply the LoRA (Hu et al., 2021) method with rank equal to 8 and scaling factor equal to 32. Due to resource constraints, we do not fine-tune Llama-3.3.

[3] huggingface.co/meta-llama/Llama-3.3-70B-Instruct
[4] huggingface.co/meta-llama/Llama-3.1-8B-Instruct

4.3 Evaluation

We assess the quality of the Causal Graphs generated by the LLMs using several evaluation metrics. The main metric is Structural Hamming Distance (SHD), a widely used measure for evaluating graph discovery algorithms (Tsamardinos et al., 2006). Lower SHD values indicate higher-quality graphs. SHD is calculated as the total number of operations—addition, removal, or reversal of edge directions—required to transform the learned graph into the target graph. Incorrectly oriented edges, where the cause and effect are reversed, are penalised as two errors. To make comparisons more meaningful across CGs of varying sizes, we report the SHD normalised by the edges count in the actual CG. We used causal discovery toolbox[5] for SHD calculations. For a more detailed analysis of selected cases, we use two additional metrics: false positives (FP), representing extra edges in the learned graph that need to be removed, and false negatives (FN), indicating missing edges that must be added to match the real graph structure. We also normalise FP and FN by true edge count in the corresponding CG.

4.4 Contamination

Even though LLMs demonstrate impressive performance on various causal tasks (Tu et al., 2023), it is crucial to understand their limitations (Tamkin et al., 2021). In the context of CGD, knowledge about the CG structure of certain CGs may have been acquired by an LLM during its training, leading to data contamination and an artificial improvement in task performance (Sainz et al., 2023).

To address this, we employ the technique proposed in Babakov et al. (2024), which provides a straightforward approach for assessing contamination. First, we prompt the target LLM to generate the list of nodes contained in the CG just based on the CG source (website and/or paper). If the risk of contamination appears high—specifically, if the number of generated nodes is close to or equal to the actual number of nodes in the BN, and the recall is close to 1—we further prompt the target LLM to construct the structure of the CG using the generated nodes. The exact prompts used for this task are detailed in the Appendix E.

methods	**GPT-4o**	**Llama-3.3**	**Llama-3.1**
	up to 10 nodes in CG		
pair	1.67	1.64	1.76
triplet	2.02	2.08	1.87
efficient	1.16	1.05	1.59
baseline	0.65	0.81	1.68
harn	0.66	0.79	1.11
delphi	0.80	0.98	0.70
finetune	0.64		0.80
up to 50 nodes in CG			
efficient	1.66	1.72	2.52
baseline	0.96	1.11	2.28
harn	0.93	1.17	1.32
delphi	1.07	1.18	1.09
finetune	1.02		1.5
all CGs			
baseline	1.0	1.14	2.21
harn	0.96	1.22	1.29
delphi	1.06	1.17	1.09
finetune	1.10		1.43

Table 2: Results of the experiments represented as SHD normalized by the real edge count. The *underscored* values indicate the method with the lowest mean SHD for each LLM within a given CG size category (i.e. the underscore is applicable for one column within a certain CG size box), as well as any methods for which the Tukey HSD test determined no statistically significant difference from the method with the lowest SHD.

5 Results

5.1 Causal Graph discovery

The results of the experiments are presented in two tables. Table 2 reports the SHD averaged across all benchmark CGs, for each method and engaged LLM. Table 3 provides a more detailed analysis of the methods used with the best-performing GPT-4o. Both tables are divided into three parts based on CG size: up to 10 nodes, up to 50 nodes, and all CGs. This division reflects the varying applicability of methods to different scopes. Specifically, the *pair* and *triplet* methods are applied only to CGs with up to 10 nodes, while the *efficient* method is used for CGs with up to 10 and 50 nodes. All other methods are applied across the full set of CGs. To study the statistical significance of the SHD difference in certain scope (i.e. for the methods used with given LLM within given CGs size) we first use the ANOVA test to check whether the group has

[5] https://github.com/ElementAI/causal_discovery_toolbox

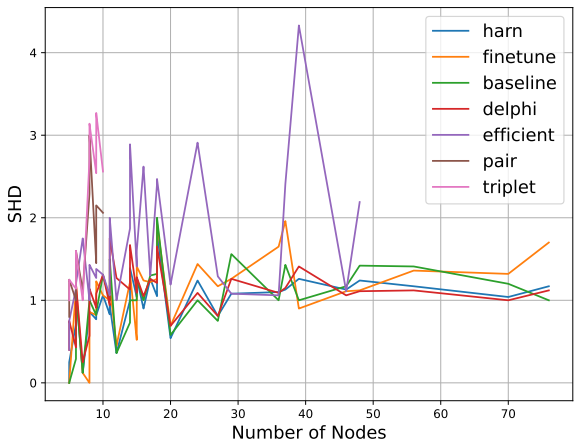

Figure 4: SHD normalized by edges count related to the number of edges in a CG.

at least one value that is statistically different from others, and if the p-value was less than 0.05 we also run Tukey HSD test to clarify which exact value is different.

The results indicate that the *pair*, *triplet*, and *efficient* methods performed worse than other approaches within their respective CG size scopes. In the evaluation across all CGs, all engaged methods showed similar performance for GPT-4o and Llama-3.3, with ANOVA tests returning p-values of 0.51 and 0.81, respectively, indicating no statistically significant differences between the methods in these scopes. For Llama-3.1, while the *baseline* method showed a statistically significant difference from others, all other methods performed similarly, with no statistically significant differences observed in pairwise comparisons using the Tukey HSD test.

Table 3 highlights the shortcomings of the *pair* and *triplet* methods for GPT-4o, with FP being notably high at 1.49 and 2.04, respectively. In contrast, the FP rates for other methods remain below 0.61. Similarly, the FP rate for the *efficient* is significantly higher at 1.03, while all other methods maintain FP rates below 0.52.

Finally, Figure 4 illustrates the dynamics of SHD as a function of the number of nodes in a CG for different methods using GPT-4o. The visualization aligns with the previous analysis, showing that the *pair* and *triplet* methods yield significantly higher SHD values within their experimental scope (CGs with up to 10 nodes). Similarly, the *efficient* method produces higher SHD compared to other methods within its scope (CGs with up to 50 nodes). The observed trend suggests that extending these methods to larger CGs is unlikely to result in improved outcomes, given their current limitations. In contrast, for the other methods, SHD values remain relatively stable even as CG size increases.

methods	**FP/edg**	**FN/edg**	**SHD/edg**
	up to 10 nodes in CG		
pair	1.49	0.18	1.67
triplet	2.04	0.1	2.02
efficient	0.61	0.58	1.16
baseline	0.26	0.39	0.65
harn	0.29	0.37	0.66
delphi	0.44	0.36	0.8
finetune	0.36	0.29	0.64
up to 50 nodes in CG			
efficient	1.03	0.65	1.66
baseline	0.33	0.64	0.96
harn	0.33	0.6	0.93
delphi	0.47	0.6	1.07
finetune	0.52	0.5	1.02
all CGs			
baseline	0.33	0.68	1.0
harn	0.31	0.65	0.96
delphi	0.41	0.65	1.06
finetune	0.56	0.56	1.1

Table 3: Detailed information about the performance of engaged methods with GPT-4o. FP/edg and FN/edg correspond to false positive and false negative edges count normalized by the true number of edges in the extracted Causal Graphs

BN	**True#**	**GPT-4o**		**Llama-3.3**		**Llama-3.1**	
		#	**Rec**	**#**	**Rec**	**#**	**Rec**
cancer	5	5	1.0	5	1.0	2	0.2
asiam	7	8	1.0	9	1.0	2	0.14
alarm	37	35	0.95	81	0.27	1	0.0

Table 4: The results of data contamination experiments on the CG nodes level. # and Rec indicate the number of nodes and Recall correspondingly.

5.2 Contamination

The results of the data contamination experiments are presented in Table 4, which highlights cases where contamination was clearly identified. The complete results for all CGs and LLMs are provided in Appendix Table 9. For each CG, we report the number of nodes generated by the LLM in terms of the contamination evaluation defined and the recall of these nodes relative to the real nodes in the CG.

LLM	CG	True edges	Gen edges	F-score	SHD
GPT-4o	asiam	8	4	0.81	0.5
GPT-4o	cancer	4	4	1.0	0.0
GPT-4o	alarm	46	46	0.63	1.43
Llama-3.3	asiam	8	4	0.81	0.5
Llama-3.3	cancer	4	4	1.0	0.0

Table 5: The results of data contamination experiments on the CG edges level.

LLMs	GPT-4o		Llama-3.3
# nodes	0-10	35-55	0-10
contaminated	0.06	1.43	0.06
non-contaminated	0.8	0.98	0.99

Table 6: Effect of data contamination reported with SHD yield by baseline method for the CGs within the same number of nodes with and without evidence of data contamination for corresponding LLM.

A CG is considered to be at high risk of contamination for a specific LLM if the number of generated nodes is close to the real number and if at the same time, the meaning of the generated nodes correspond to the majority of the real nodes. Thus, we select the following thresholds: less than 15% deviation from the actual number of nodes in the CG, and a recall of more than 0.85.

In Table 4, we observe that the *cancer* and *asiam*[6] CGs are known to both GPT-4o and `Llama-3.3`. Additionally, GPT-4o has also clearly encountered the *alarm* CG during its training.

Table 5 shows the experiments of prompting LLMs to generated the exact structure of the CGs which are counted as high risk of contamination. The results confirm the contamination of the *cancer* CG for both LLMs, as the generated structures closely match the real one. The *asiam* CG is also likely known to both LLMs, albeit with a slightly higher number of structural inaccuracies compared to *cancer*. In the case of *alarm*, although GPT-4o has seen the CG during training, it has not successfully learned its structure, as the generated graph deviates significantly from the actual one.

Even though several CGs were identified as contaminated, the critical question is whether this contamination significantly affects the performance of the engaged methods using these LLMs. To address this, Table 6 compares the SHD produced by the *baseline* method with GPT-4o for CGs with and without evidence of contamination, grouped by size. The *baseline* method was chosen because it is conceptually closest to the setup used for contamination checks, with a slight modification: while the contamination check required recreating nodes and edges based only on source references, the *baseline* method includes only CG idea, knowledge area, and clarified node names, which differ slightly from those in the source.

The results show that for small CGs (up to 10 nodes), contamination has a noticeable effect on performance - both LLMs applied to contaminated CGs resulted in significantly lower SHD compared with non-contaminated ones. However, for larger CGs, contamination appears to have no substantial impact, because SHD for the *alarm* CG with GPT-4o is even higher than that for other CGs of similar size.

6 Discussion

Our benchmark enabled the first direct comparison of numerous LLM-based CGD methods, providing for the first time a standardized evaluation framework that was previously lacking in this scientific area. This allows for a more objective assessment of different approaches under the same conditions. In this section, we analyze the results of the experiments, explore the challenges associated with applying specific LLMs and methods, and extract key insights gained from this unified comparison.

The task of CGD proves to be demanding in terms of LLM capabilities, as evidenced by the consistent decline in performance with smaller LLMs, regardless of the method applied. Additionally, the more complex the method, the higher the requirements for LLM capabilities, particularly in scenarios where the queries to the LLM depend on the accurate parsing of results from previous calls.

In our experiments, this limitation became apparent when using `Llama-3.1` with methods like *harn*. After the revision step, the method expects the list of edges in a format that can be automatically processed to remove incorrect edges. However, `Llama-3.1` frequently failed to generate outputs in the required format, leading to parsing errors and hindering further automation.

Even less complex methods that require a high number of queries, such as *pair* and *triplet*, presented challenges with `Llama-3.1`. Although the expected output for each query is relatively sim-

[6]Widely-known ASIA network (Lauritzen and Spiegelhalter, 1988) without "either" node.

ple (a small JSON), `Llama-3.1` often produced an incorrect form of JSON, necessitating manual intervention to fix the results.

Another key insight is that methods relying on exhaustive querying of all possible combinations of nodes, such as *triplet* and *pair*, along with the slightly less demanding but still query-intensive *efficient* method, tend to be ineffective despite their intuitive appeal. Their performance is consistently worse than that of other methods. The most likely explanation for this underperformance is that asking the LLM overly specific questions about a limited number of nodes may lead it to "overthink" the importance of causal links between those nodes, ignoring the global causal context of the target CG. This hyper-focus on isolated relationships results in outputs that are less aligned with the overall structure of the CG, ultimately reducing the accuracy and utility of these methods.

Methods that utilize all nodes of a CG within a single query (*baseline*, *harn*, *delphi*, and *finetune*) consistently demonstrate significantly better performance than query-intensive methods. While SHD values fluctuate across methods, statistical significance tests indicate no meaningful differences exist between them. This suggests that providing all nodes at once is an effective strategy for CGD. Furthermore, this indicates that complex querying schemes may be unnecessary. Simple approaches, such as a single prompt *baseline* or two prompts *harn* achieve comparable performance to more intricate methods like *delphi*, which requires multiple calls to different LLM-experts before merging their outputs into a final CG.

Fine-tuning LLMs for the CGD task performs on par with the best methods but does not surpass them. Since the *finetune* method essentially replicates the *baseline* with additional training on limited CGD-specific data, this result suggests the need for more extensive and diverse training data. In our training data preparation, we used only one target sequence for the generated CG. However, generating the correct list of edges in any sequence is acceptable for CGD tasks. Addressing this in future data preparation could further enhance the fine-tuning process.

A common challenge across all methods and LLMs is that performance deteriorates as the size of the CG increases. For larger graphs, SHD approaches 1 even for the best-performing methods, indicating that errors scale with the number of nodes and edges. Furthermore, we encountered an issue where even large LLMs like GPT-4o and `Llama-3.3`, struggled to generate a complete list of edges when dealing with a large number of nodes because of the limit of the generated tokens. This suggests that LLM-driven CGD is best suited for graphs with limited nodes.

Our results also show that, even though CGs are rarely explicitly described in training data (because of their graphical nature), some LLMs have clearly encountered certain CGs during pre-training. This highlights the need for preliminary contamination checks for each CG and LLM pair before conducting experiments. If contamination is detected, the affected CG could be excluded from further experiments with that LLM, or alternatively, node names could be paraphrased to reduce the likelihood of contamination. However, paraphrasing node names introduces a risk of altering their semantic meaning, which may compromise the fairness of the evaluation by providing the LLM with corrupted information about the nodes forming a CG.

7 Future work

Our current experiments evaluate LLM-based methods using only the names of CG nodes, without incorporating the underlying data or comparing results to classical structure learning algorithms. In future work, we plan to extend the benchmark by including experiments with traditional data-driven causal discovery methods, as well as hybrid approaches that combine LLM-driven and data-driven techniques. This will provide a more comprehensive assessment of the relative strengths and weaknesses of LLMs in causal graph discovery and help clarify their utility alongside established approaches.

Additionally, our evaluation focused primarily on GPT-4o and Llama-series models. Exploring a broader range of language models, including both proprietary and open-source variants, in future studies could provide a more robust and generalizable understanding of LLM capabilities in causal graph discovery.

8 Conclusion

In this paper, we introduced CausalGraphBench, a benchmark specifically designed to evaluate the capabilities of LLMs in constructing Causal Graphs. The benchmark consists of 35 Causal Graphs sourced from publicly available repositories and academic papers, accompanied by detailed

metadata to facilitate systematic evaluation. Our results demonstrate that the benchmark provides a valuable framework for assessing LLM-driven Causal Graph Discovery methods, enabling a direct comparison of numerous approaches under standardized conditions—a comparison that, to our knowledge, had not been conducted before. We assessed several diverse methods using this benchmark, ranging from simple single-query approaches to more complex, multi-step, and query-intensive methods. Additionally, we explored the effects of data contamination on the performance of the models, further validating the benchmark as a helpful tool for advancing research in this area.

Our results reveal several key insights. Methods that leverage all nodes of the Causal Graph in a single query demonstrate superior performance, particularly when they incorporate iterative refinement or rely on minimal query complexity. By contrast, methods that perform exhaustive queries, such as evaluating all node pairs or triplets, tend to underperform, likely due to over-focusing on local relationships at the expense of the broader graph context. Across all methods and LLMs, performance decreased as graph size increased, emphasizing scalability as a persistent challenge. Future research could focus on scalable solutions, such as processing smaller graph clusters sequentially and merging results.

Limitations

Our study has certain limitations that should be acknowledged. First, we used a basic implementation of the pairwise querying method without incorporating additional techniques proposed in various papers, which might affect its comparative performance. Second, there is a slight possibility of errors or misunderstandings in our reproduction of methods from other researchers, despite our best efforts to remain faithful to their descriptions.

To address these limitations and foster further research, we will make the benchmark available. This will enable future Causal Graph Discovery methods to be applied to our benchmark, evaluated using standardized tools, and their results integrated into the public metrics table, ensuring transparency and facilitating the continued development of this field.

Moreover, as part of our benchmark construction, a significant number of causal graphs were manually extracted from figures in academic papers. This reliance on visually available graphs may introduce some degree of selection bias, which could affect the representativeness of the benchmark and, consequently, the generalizability of the results.

Acknowledgments

This paper is part of the R+D+i project TED2021-130295B-C33, funded by MCIN/AEI/10.13039/501100011033/ and by the "European Union NextGenerationEU/PRTR". This research also contributes to the projects PID2020-112623GB-I00 and PID2023-149549NB-I00 funded by MCIN/AEI/10.13039/501100011033/ and by ERDF A way of making Europe. The support of the Galician Ministry for Education, Universities and Professional Training and the "ERDF A way of making Europe" is also acknowledged through grants "Centro de investigación de Galicia accreditation 2024-2027 ED431G-2023/04" and "Reference Competitive Group accreditation 2022-2025 ED431C 2022/19"

References

Nikolay Babakov, Ehud Reiter, and Alberto Bugarín-Diz. 2024. Scalability of Bayesian Network structure elicitation with Large Language Models: a novel methodology and comparative analysis. *arXiv preprint arXiv:2407.09311*.

Nikolay Babakov, Adarsa Sivaprasad, Ehud Reiter, and Alberto Bugarín-Diz. 2025. Reusability of Bayesian Networks case studies: a survey. *Applied Intelligence*, 55(6):417.

Taiyu Ban, Lyuzhou Chen, Derui Lyu, Xiangyu Wang, and Huanhuan Chen. 2023a. Causal structure learning supervised by Large Language Model. *arXiv preprint arXiv:2311.11689*.

Taiyu Ban, Lyvzhou Chen, Xiangyu Wang, and Huanhuan Chen. 2023b. From query tools to causal architects: Harnessing Large Language Models for advanced causal discovery from data. *arXiv preprint arXiv:2306.16902*.

Cedric Baudrit, Patrice Buche, Nadine Leconte, Christophe Fernandez, Maëllis Belna, and Geneviève Gésan-Guiziou. 2022. Decision support tool for the agri-food sector using data annotated by ontology and Bayesian Network: A proof of concept applied to milk microfiltration. *International Journal of Agricultural and Environmental Information Systems (IJAEIS)*, 13(1):1–22.

Sirui Chen, Mengying Xu, Kun Wang, Xingyu Zeng, Rui Zhao, Shengjie Zhao, and Chaochao Lu. 2024. CLEAR: Can language models really understand causal graphs? In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages

6247–6265, Miami, Florida, USA. Association for Computational Linguistics.

Sabarathinam Chockalingam, Wolter Pieters, André Teixeira, Nima Khakzad, and Pieter Van Gelder. 2019. Combining Bayesian Networks and fishbone diagrams to distinguish between intentional attacks and accidental technical failures. In *Graphical Models for Security: 5th International Workshop, GraMSec 2018, Oxford, UK, July 8, 2018, Revised Selected Papers 5*, pages 31–50. Springer.

Kai-Hendrik Cohrs, Gherardo Varando, Emiliano Diaz, Vasileios Sitokonstantinou, and Gustau Camps-Valls. 2024. Large Language Models for constrained-based causal discovery. *arXiv preprint arXiv:2406.07378*.

Victor-Alexandru Darvariu, Stephen Hailes, and Mirco Musolesi. 2024. Large Language Models are effective priors for causal graph discovery. *arXiv preprint arXiv:2405.13551*.

Tao Feng, Lizhen Qu, Niket Tandon, Zhuang Li, Xiaoxi Kang, and Gholamreza Haffari. 2024. From pretraining corpora to Large Language Models: What factors influence LLM performance in causal discovery tasks? *arXiv preprint arXiv:2407.19638*.

Daya Guo, Qihao Zhu, Dejian Yang, Zhenda Xie, Kai Dong, Wentao Zhang, Guanting Chen, Xiao Bi, Yu Wu, YK Li, et al. 2024. Deepseek-coder: When the Large Language Model meets programming–the rise of code intelligence. *arXiv preprint arXiv:2401.14196*.

Hideki Hamayasu, Masashi Miyao, Chihiro Kawai, Toshio Osamura, Akira Yamamoto, Hirozo Minami, Hitoshi Abiru, Keiji Tamaki, and Hirokazu Kotani. 2022. A proof-of-concept study to construct Bayesian Network decision models for supporting the categorization of sudden unexpected infant death. *Scientific Reports*, 12(1):9773.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of Large Language Models. *arXiv preprint arXiv:2106.09685*.

Liting Jing, Bowen Tan, Shaofei Jiang, and Junfeng Ma. 2021. Additive manufacturing industrial adaptability analysis using fuzzy Bayesian Network. *Computers & Industrial Engineering*, 155:107216.

Thomas Jiralerspong, Xiaoyin Chen, Yash More, Vedant Shah, and Yoshua Bengio. 2024. Efficient causal graph discovery using Large Language Models. *arXiv preprint arXiv:2402.01207*.

Enkelejda Kasneci, Kathrin Sessler, Stefan Küchemann, Maria Bannert, Daryna Dementieva, Frank Fischer, Urs Gasser, Georg Groh, Stephan Günnemann, Eyke Hüllermeier, Stephan Krusche, Gitta Kutyniok, Tilman Michaeli, Claudia Nerdel, Jürgen Pfeffer, Oleksandra Poquet, Michael Sailer, Albrecht Schmidt, Tina Seidel, Matthias Stadler, Jochen Weller, Jochen Kuhn, and Gjergji Kasneci. 2023. ChatGPT for good? on opportunities and challenges of Large Language Models for education. *Learning and Individual Differences*, 103:102274.

Ha Duy Khanh, Soo Yong Kim, et al. 2022. Construction productivity prediction through Bayesian Networks for building projects: Case from vietnam. *Engineering, Construction and Architectural Management*, 30(5):2075–2100.

Emre Kıcıman, Robert Ness, Amit Sharma, and Chenhao Tan. 2023. Causal reasoning and Large Language Models: Opening a new frontier for causality. *arXiv preprint arXiv:2305.00050*.

Neville Kenneth Kitson, Anthony C Constantinou, Zhigao Guo, Yang Liu, and Kiattikun Chobtham. 2023. A survey of Bayesian Network structure learning. *Artificial Intelligence Review*, pages 1–94.

Daphane Koller. 2009. Probabilistic graphical models: Principles and techniques.

Kevin B Korb and Ann E Nicholson. 2010. *Bayesian artificial intelligence*. CRC press.

Steffen L Lauritzen and David J Spiegelhalter. 1988. Local computations with probabilities on graphical structures and their application to expert systems. *Journal of the Royal Statistical Society: Series B (Methodological)*, 50(2):157–194.

Yunpeng Liu, Shen Wang, Qian Liu, Dongpeng Liu, Yang Yang, Yong Dan, and Wei Wu. 2022. Failure risk assessment of coal gasifier based on the integration of Bayesian Network and trapezoidal intuitionistic fuzzy number-based similarity aggregation method (tpifn-sam). *Processes*, 10(9):1863.

Stephanie Long, Alexandre Piché, Valentina Zantedeschi, Tibor Schuster, and Alexandre Drouin. 2023a. Causal discovery with language models as imperfect experts. *arXiv preprint arXiv:2307.02390*.

Stephanie Long, Tibor Schuster, and Alexandre Piché. 2023b. Can Large Language Models build causal graphs? *arXiv preprint arXiv:2303.05279*.

Volodymyr Lytvynenko, Oleksandr Naumov, Mariia Voronenko, Jan Krejci, Larisa Naumova, Dmytro Nikytenko, and Nataliia Savina. 2020. Dynamic Bayesian Networks application for evaluating the investment projects effectiveness. In *International Scientific Conference "Intellectual Systems of Decision Making and Problem of Computational Intelligence"*, pages 315–330. Springer.

Eugenio Molina-Navarro, Pedro Segurado, Paulo Branco, Carina Almeida, and Hans E Andersen. 2020. Predicting the ecological status of rivers and streams under different climatic and socioeconomic scenarios using Bayesian belief Networks. *Limnologica*, 80:125742.

Mariana R Neves, Bridget J Daley, Graham A Hitman, Mohammed SB Huda, Scott McLachlan, Sarah Finer, and William Marsh. 2021. Causal dynamic Bayesian Networks for the management of glucose control in gestational diabetes. In *2021 IEEE 9th International Conference on Healthcare Informatics (ICHI)*, pages 31–40. IEEE.

Rachael H Nolan, Jennifer Sinclair, Cathleen M Waters, Patrick J Mitchell, David J Eldridge, Keryn I Paul, Stephen Roxburgh, Don W Butler, and Daniel Ramp. 2019. Risks to carbon dynamics in semi-arid woodlands of eastern australia under current and future climates. *Journal of Environmental Management*, 235:500–510.

Erik P. Nyberg, Ann E. Nicholson, Kevin B. Korb, Michael Wybrow, Ingrid Zukerman, Steven Mascaro, Shreshth Thakur, Abraham Oshni Alvandi, Jeff Riley, Ross Pearson, Shane Morris, Matthieu Herrmann, A.K.M. Azad, Fergus Bolger, Ulrike Hahn, and David Lagnado. 2022. BARD: A structured technique for group elicitation of Bayesian Networks to support analytic reasoning. *Risk Analysis*, 42(6):1155–1178.

Helder CR Oliveira, Svetlana Yanushkevich, and Mohammed Almekhlafi. 2022. Sensitivity analysis of stroke predictors using structural equation modeling and Bayesian Networks. In *2022 IEEE Conference on Computational Intelligence in Bioinformatics and Computational Biology (CIBCB)*, pages 1–8. IEEE.

Vassilis Panopoulos, Apostolos Bougas, Borja Garcia de Soto, and Bryan T Adey. 2021. Using Bayesian Networks to estimate bridge characteristics in early road designs. *Infrastructure Asset Management*, 9(1):40–56.

Yunita Rachma Pradiawati, Yanti Rusmawati, and Muhammad Arzaki. 2019. Reasoning about the disruption patterns for train system using Bayesian Network and prolog. In *Journal of Physics: Conference Series*, volume 1192, page 012064. IOP Publishing.

Nurulhuda Ramli, Noraida Abdul Ghani, Nazihah Ahmad, and Intan Hashimah Mohd Hashim. 2021. Psychological response in fire: a fuzzy Bayesian Network approach using expert judgment. *Fire technology*, 57:2305–2338.

Jessica A Ramsay, Steven Mascaro, Anita J Campbell, David A Foley, Ariel O Mace, Paul Ingram, Meredith L Borland, Christopher C Blyth, Nicholas G Larkins, Tim Robertson, et al. 2022. Urinary tract infections in children: building a causal model-based decision support tool for diagnosis with domain knowledge and prospective data. *BMC Medical Research Methodology*, 22(1):218.

Oscar Sainz, Jon Ander Campos, Iker García-Ferrero, Julen Etxaniz, Oier Lopez de Lacalle, and Eneko Agirre. 2023. NLP evaluation in trouble: On the need to measure LLM data contamination for each benchmark. *arXiv preprint arXiv:2310.18018*.

Lydie Samie, Christophe Champod, Séverine Delémont, Patrick Basset, Tacha Hicks, and Vincent Castella. 2022. Use of Bayesian Networks for the investigation of the nature of biological material in casework. *Forensic Science International*, 331:111174.

Gabriele Sottocornola, Sanja Baric, Fabio Stella, and Markus Zanker. 2023. Development of a knowledge-based expert system for diagnosing post-harvest diseases of apple. *Agriculture*, 13(1):177.

Alex Tamkin, Miles Brundage, Jack Clark, and Deep Ganguli. 2021. Understanding the capabilities, limitations, and societal impact of Large Language Models. *arXiv preprint arXiv:2102.02503*.

Ioannis Tsamardinos, Laura E Brown, and Constantin F Aliferis. 2006. The max-min hill-climbing Bayesian Network structure learning algorithm. *Machine learning*, 65:31–78.

Ruibo Tu, Chao Ma, and Cheng Zhang. 2023. Causal-discovery performance of chatgpt in the context of neuropathic pain diagnosis. *arXiv preprint arXiv:2301.13819*.

Aniket Vashishtha, Abbavaram Gowtham Reddy, Abhinav Kumar, Saketh Bachu, Vineeth N Balasubramanian, and Amit Sharma. 2023. Causal inference using LLM-guided discovery. *arXiv preprint arXiv:2310.15117*.

Hana Catur Wahyuni, Iwan Vanany, and Udisubakti Ciptomulyono. 2019. Application of Bayesian Network for food safety risk in cattle slaugtering industry. In *2019 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM)*, pages 450–454. IEEE.

Guangya Wan, Yuqi Wu, Mengxuan Hu, Zhixuan Chu, and Sheng Li. 2024. Bridging causal discovery and Large Language Models: A comprehensive survey of integrative approaches and future directions. *arXiv preprint arXiv:2402.11068*.

Heng Wang, Shangbin Feng, Tianxing He, Zhaoxuan Tan, Xiaochuang Han, and Yulia Tsvetkov. 2024. Can language models solve graph problems in natural language? *Advances in Neural Information Processing Systems*, 36.

Deng Xiaoyong, Qiu Siyu, Zhang Dongyang, and Jin Xueke. 2021. Vulnerability assessment based on fuzzy Bayesian Network. In *2021 2nd International Conference on Electronics, Communications and Information Technology (CECIT)*, pages 1283–1287. IEEE.

Qin Zhang, Jiabin Chen, Jiaxing Sun, and Junyu Liang. 2021. Soldier threat assessment method based on Bayesian Network. In *2021 China Automation Congress (CAC)*, pages 1750–1755. IEEE.

Yuzhe Zhang, Yipeng Zhang, Yidong Gan, Lina Yao, and Chen Wang. 2024. Causal graph discovery with retrieval-augmented generation based Large Language Models. *arXiv preprint arXiv:2402.15301*.

Yu Zhou, Xingyu Wu, Beicheng Huang, Jibin Wu, Liang Feng, and Kay Chen Tan. 2024. Causalbench: A comprehensive benchmark for causal learning capability of Large Language Models. *arXiv preprint arXiv:2404.06349*.

Enrico Zio, Maryam Mustafayeva, and Andrea Montanaro. 2022. A Bayesian belief Network model for the risk assessment and management of premature screen-out during hydraulic fracturing. *Reliability Engineering & System Safety*, 218:108094.

A Benchmark information

paper / CG	Ban et al. (2023b)	Babakov et al. (2024)	Jiralerspong et al. (2024)	Vashishtha et al. (2023)	Long et al. (2023b)	Cohrs et al. (2024)	Darvariu et al. (2024)	Zhou et al. (2024)
cancer	✓			✓	✓	✓		✓
burglary						✓		
asia	✓		✓	✓		✓	✓	✓
earthquake				✓				✓
child	✓		✓	✓			✓	✓
alarm	✓	✓						
insurance	✓	✓					✓	✓
water	✓							✓
mildew	✓							✓
sachs						✓		✓
barley	✓	✓						✓
hailfinder		✓						✓
pathfinder		✓						
andes		✓						
diabetes		✓			✓			
munin		✓						
hepar2		✓						✓
survey				✓				✓
win95								✓
coma		✓						
covid		✓						
agro		✓						
sperm		✓						
screen		✓						
sids		✓						
apple		✓						
urinary		✓						
spurious						✓		
bk-spv						✓		
nao-dk						✓		
neuropatic			✓	✓	✓			
alcohol						✓		
obesity					✓			

Table 7: Overview of the CGs used in the different papers introducing the LLMs application for Causal Graph construction.

CG name	Source	# nodes	# edges	Pub. avail.
agro	(Baudrit et al., 2022)	6	10	×
stroke	(Oliveira et al., 2022)	6	7	×
attack_failure	(Chockalingam et al., 2019)	8	7	×
aircraft_vulnerability	(Xiaoyong et al., 2021)	8	7	×
sperm_criminal	(Samie et al., 2022)	9	11	×
bridge	(Panopoulos et al., 2021)	9	13	×
carbon_risks	(Nolan et al., 2019)	10	16	×
response_in_fire	(Ramli et al., 2021)	11	12	×
food_safety	(Wahyuni et al., 2019)	12	11	×
glucose_control	(Neves et al., 2021)	14	18	×
train_disruption	(Pradiawati et al., 2019)	14	15	×
investment	(Lytvynenko et al., 2020)	15	22	×
river_status	(Molina-Navarro et al., 2020)	15	25	×
screen_out	(Zio et al., 2022)	16	21	×
sids	(Hamayasu et al., 2022)	17	27	×
construction_productivity	(Khanh et al., 2022)	18	19	×
soldier_threat	(Zhang et al., 2021)	18	17	×
additive_manufacturing	(Jing et al., 2021)	24	34	×
apple	(Sottocornola et al., 2023)	29	62	×
urinary	(Ramsay et al., 2022)	36	107	×
coal_gasifier_risk	(Liu et al., 2022)	39	39	×
cancer	bnlearn	5	4	✓
coma	bayesfusion	5	5	✓
asiam	bnlearn, (Lauritzen and Spiegelhalter, 1988)	7	8	✓
sachs	bnlearn	11	17	✓
covid	bayesfusion	20	26	✓
insurance	bnlearn	27	52	✓
alarm	bnlearn	37	46	✓
ecoli70	bnlearn	46	70	✓
barley	bnlearn	48	84	✓
hailfinder	bnlearn	56	66	✓
hepar2	bnlearn	70	123	✓
win95pts	bnlearn	76	112	✓
munin1	bnlearn	186	273	✓
neuro	bnlearn	222	770	✓

Table 8: Full list of the CGs forming the CausalGraphBench.

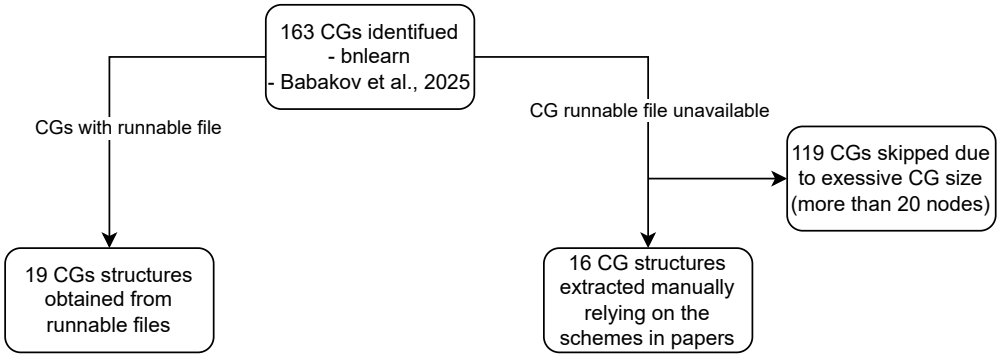

Figure 5: The diagram of the selection of CGs for the benchmark.

B Metadata extraction process for Causal Graphs description

B.1 System message

Consider the following paper describing the Bayesian Network *title of the paper*.

The paper's content: *paper content*

The node names of the Causal Graph of the presented Bayesian Network is: *list of names of the nodes and their corresponding states (if applicable)*

B.2 Prompt for "CG idea"

Describe the purpose of the Bayesian Network presented in the paper. Describe it in 2-3 sentences.

B.3 Prompt for "Knowledge area"

Describe the area of knowledge necessary to understand the paper (only domain areas are needed, without probability theory etc.). Just list the topics separated by a comma.

B.4 Prompt for "Nodes dict"

Consider the node names of the Causal Graph of the presented Bayesian Network: *list of names of the nodes and their corresponding states (if applicable)*

Your task is to understand the real meaning of each variable according to their possible values, and explain them in order.

Return the JSON dictionary of form "original node name" : "clarified node name"

C Example of the extracted metadata

In this section, we provide an example of the full metadata extracted from the paper describing the bridge CG (Panopoulos et al., 2021).

CG idea: The Bayesian Network (BN) described is used to estimate bridge characteristics in the early stages of road design projects. It leverages available situational information (e.g., span length, number of spans) to predict key design elements of bridges, such as bridge type, cross-section, and superstructure depth. This method allows decision-makers to quickly and reliably predict bridge features with limited initial data, improving the accuracy and efficiency of early project planning

Knowledge area: Civil engineering, bridge design, infrastructure planning, construction management, road design, structural engineering, transportation infrastructure.

Nodes dictionary:

"Bridge_length": "Bridge total length", "Span_No": "Number of spans", "Underneath": "Object under bridge", "Pier_height": "Height of piers", "Operation": "Bridge function", "Max_span_length": "Maximum span length", "Bridge_type": "Type of bridge", "Cross_section_depth": "Depth of cross-section", "Cross_section_type": "Type of cross-section"

Structure:

["Bridge_length", "Pier_height"], ["Bridge_length", "Underneath"], ["Bridge_length", "Span_No"], ["Bridge_length", "Max_span_length"], ["Bridge_length", "Cross_section_type"], ["Span_No", "Underneath"], ["Span_No", "Bridge_type"], ["Span_No", "Max_span_length"], ["Underneath", "Pier_height"], ["Underneath", "Operation"], ["Max_span_length", "Bridge_type"], ["Max_span_length", "Cross_section_depth"], ["Bridge_type", "Cross_section_type"]

Source: "paper": "Using Bayesian networks to estimate bridge characteristics in early road designs"

D Causal discovery methods

D.1 Prompt of "baseline" method

You are an expert on *knowledge area*. You are constructing the Bayesian Network aimed to fulfill the following task: *CG idea*. To construct the Bayesian Network you need to investigate the cause-and-effect relationships between the following variables in your area of expertise: *clarified node names*. Based on the meaning of variables, analyze the cause-and-effect relationships between them. Please give the results as a directed graph network. Make sure that each edge represent a direct causality between the two variables.

Return valid JSON-list of the following format:

from node (A), to node(B), # (meaning that there is a direct causal effect from node A to node B)

from node (F), to node(E)) # (meaning that there is a direct causal effect from node F to node E)

...

Obligatory return just list of node pairs representing causal relations, no dictionaries or other formats

D.2 Prompt of "harn" method (used after "Baseline" prompt)

Based on your explanation, check whether the following causal relations are correct, and give the reasons. (Recall that the notation "['Node A', 'Node B']" means that there is direct causal effect of Node A to Node B): *causal structure from the baseline prompt*

Return valid JSON that will contain only invalid causal statements in the following format:

'from node (A)', 'to node(B)': "Explanation why there is NO causal effect from node A to node B ...

If you consider all causal statements to be correct, return an empty JSON.

D.3 Prompt of "pair" method

You are an expert on *knowledge area*. You are constructing the Causal Graph aimed to fulfill the following task: *CG idea*.

Consider the following factors related to the task of the Causal Graph which can have various causal effects on each other. *factor A*

factor B

There are three possible relationships between *factor A* and *factor B*:

A. Changing the value of *factor A* will cause a change in *factor B*. B. Changing the value of *factor B* will cause a change in *factor A*. C. There is no causal relationship between *factor A* and *factor B*.

Think step by step. Then, provide your final answer (variable names only) in the form of a valid JSON-list of the following format: "'json

factor A, *factor B* meaning that there is a direct causal effect from node *factor A* to *factor B*

factor B, *factor A* meaning that there is a direct causal effect from *factor B* to node *factor A*

[] meaning that there is no direct causal effect between the two nodes

You must return only one of the three options. Return obligatory list (not other data structures) and keep the naming of the variables as in the input data.

D.4 Prompt of "triplet" method

You are an expert on *knowledge area*. You are constructing the Causal Graph aimed to fulfill the following task: *CG idea*.

Identify the causal relationships between the given variables and create a directed acyclic graph. Make sure to give a reasoning for your answer and then output the directed graph in the form of a

list of tuples, where each tuple is a directed edge. The desired output should be in the following form: $[(\text{"}A\text{"}, \text{"}B\text{"}], (\text{"}B\text{"}, \text{"}C\text{"}]]$ where first tuple represents a directed edge from Node "A" to Node "B", second tuple represents a directed edge from Node "B" to Node "C"and so on. If a node should not form any causal relationship with other nodes, then you can add it as an isolated node of the graph by adding it seperately. For example, if "C" should be an isolated node in a graph with nodes "A", "B", "C", then the final DAG representation should be like $[(\text{"}A\text{"}, \text{"}B\text{"}], [\text{"}C\text{"}]]$. Use the description about the node provided with the nodes in brackets to form a better decision about the causal direction orientation between the nodes.

Example: Input: Nodes: $[\text{"}A\text{"}, \text{"}B\text{"}, \text{"}C\text{"}]$; Return a valid JSON of the following format: Output: $[[\text{"}A\text{"}, \text{"}B\text{"}], (\text{"}B\text{"}, \text{"}C\text{"}]]$ meaning that A causes B and B causes C

$[[\text{"}A\text{"}, \text{"}B\text{"}], [\text{"}C\text{"}]]$ meaning that A causes B and C is an isolated node

$[[\text{"}A\text{"}, \text{"}C\text{"}], [\text{"}B\text{"}, \text{"}C\text{"}]]$ meaning that A causes C and B causes C

sub

D.5 Prompts of "efficient" method

Querying independent nodes

You are an expert on *knowledge area*. You are constructing the Causal Graph aimed to fulfill the following task: *CG idea*.

The following factors are key variables related to the task of the Causal Graph which have various causal effects on each other. Our goal is to construct a Causal Graph between these variables: *clarified node names*

Now you are going to use the data to construct a Causal Graph. You will start with identifying the variable(s) that are unaffected by any other variables. Think step by step.

Then, provide your final answer (variable names only) as valid JSON-list of the following format: $[node(A), node(B), ...]$

Querying the rest of nodes

Given that the following varibales <*list of independent nodes*> are not affected by any other variable and the following causal relationships (in the form $[node(A), node(B)]$, meaning that there is a direct causal effect from node to node A to node B) have been identified: *previously collected structure*.

Select the variables that are caused by <*current node*>. The variables that can be caused by <*current node*> are *potentially caused nodes*.

Think step by step. Then, provide your final answer (variable names only) in the form of valid JSON-list of the following format: $[[\text{"}nodeA\text{"}, \text{"}nodeB\text{"}], [\text{"}nodeB\text{"}, \text{"}nodeC\text{"}]...]$

If you believe that there are no variables caused by <*current node*>, return an empty JSON-list. []

D.6 Prompts of "delphi" method

D.6.1 Facilitator prompts

System message

We are going to collect a Bayesian Network using a special communication protocol. The protocol is based on the paper "BARD: A Structured Technique for Group Elicitation of Bayesian Networks to Support Analytic Reasoning". It assumes that several specialists possess the necessary skills in the Bayesian Networks problem domain, and respond to our questions independently. Then we match their responses and help them to discuss the answers in an anonymous mode if any disagreements are found until a collective agreement is achieved.

First prompt requesting to think about possible profiles of the experts

We are going to collect a Bayesian network that requires some knowledge about *knowledge area* Here is the general idea of the Bayesian Network: *CG idea*. We will use another Large Language Model as experts. We will need 9 profiles of the experts that will be used to initialize the system message of the Language Model. The profiles must be as diverse as possible but at the same time, they must jointly possess all necessary knowledge to fulfill the task of knowledge elicitation for Bayesian Network collection. Think step-by-step what are the main qualities such experts should possess.

Second prompt requesting to generate a valid JSON with the profiles of the experts

Now please propose to me 9 profiles of the experts that will be used to initialize the system message of the Language Model. Turn your answer into JSON of the following form. Obligatory use such json from and do not include any side comments “‘json {"expert_1": "textual description of expert" (simply copy paste the details you used in the previous reply), "expert_2": "textual description of expert"(simply copy paste the details you used in the previous reply), "expert_3": "textual description of expert" (simply copy paste the details you used in the previous reply) ... } “‘

D.6.2 LLM expert prompts

System message

You will generate a predictive model using a specialized communication protocol. Assume the presence of multiple specialists possessing the required skills in the designated problem domain. Each specialist responds independently to our questions. Provide input as an expert with the following profile: *profile of the expert*.

First prompt demonstrating the list of explicit names of the CG nodes and requesting to reason about possible causal relations between them.

Consider the factors associated with the predictive model, represented by the list of nodes:*list of explicit names of CG nodes*. Now, analyze the relationships between these factors.

There are three possible types of relations:

- Factor A directly affects Factor B
- Factor B directly affects Factor A
- No direct effect between the two factors

Please systematically evaluate the interconnections between the specified factors, focusing only on significant relations

Second prompt requesting to summarize the generated causal relationships into a valid JSON

Summarize your thoughts in valid JSON format based on the relationships between the specified factors: *list of explicit names of CG nodes*. Use the following format to indicate connections between factors A and B: [(factor A, factor B)] (indicating that A directly affects B). Obligatorily keep the original names of the specified factors, do not change any letter from them. Provide only the valid JSON representation without additional discussion, following this structure:

[[factor A, factor B], (meaning the factor A directly affects factor B)
[factor C, factor E], (meaning the factor C directly affects factor E)
[factor D, factor H], (meaning the factor D directly affects factor H)
..... [factor .., factor ...]]

E Prompts for assessing the contamination

E.1 Node generation prompt

Generate a list of nodes in the Bayesian Network discussed in paper "*paper*" and also available on *website*. The Bayesian Network is designed for *CG idea*. It is related to the following areas of knowledge: *knowledge areas*. Provide details on each node and its role within the network structure. Return JSON of form "node_name": "meaning of the node in the Bayesian Network"

E.2 Structure generation prompt

Now retrieve the edges connecting the previously mentioned nodes in the Bayesian Network described in paper". Express the network structure using the ’A->B’ notation, indicating the presence of an edge from node A to node B in the Bayesian Network. Return JSON of form

[from_node (A), to_node (B)], [from_node, to_node], ...

Node matching prompt (for GPT-4o)

This is the list of nodes in the Bayesian Network and their corresponding meaning

BN	True#	GPT-4o		Llama-3.3		Llama-3.1	
		#	Rec	#	Rec	#	Rec
coma	5	17	0.4	17	0.4	1	0.2
cancer	5	5	1.0	5	1.0	2	0.2
stroke	6	17	0.83	14	0.67	2	0.17
agro	6	11	0.67	12	0.33	143	0.5
asiam	7	8	1.0	9	1.0	2	0.14
attack_failure	8	15	0.25	19	0.12	1	0.0
aircraft_vulnerability	8	15	0.0	14	0.12	11	0.12
bridge	9	11	0.56	11	0.44	2	0.11
sperm_criminal	9	10	0.0	20	0.33	106	0.22
carbon_risks	10	12	0.5	12	0.9	1	0.0
sachs	11	13	1.0	10	0.82	2	0.0
response_in_fire	11	11	0.55	10	0.55	25	0.27
food_safety	12	10	0.5	13	0.67	62	0.17
glucose_control	14	15	0.5	11	0.36	1	0.07
train_disruption	14	13	0.0	13	0.14	17	0.14
investment	15	13	0.47	14	0.4	34	0.4
river_status	15	13	0.13	11	0.2	1	0.0
screen_out	16	10	0.06	17	0.19	135	0.06
sids	17	15	0.18	16	0.29	20	0.18
construction_productivity	18	15	0.33	14	0.33	1	0.0
soldier_threat	18	9	0.17	15	0.33	60	0.78
covid	20	15	0.15	14	0.2	33	0.15
additive_manufacturing	24	9	0.25	14	0.08	2	0.04
insurance	27	14	0.33	11	0.07	2	0.04
apple	29	12	0.07	15	0.1	185	0.07
urinary	36	14	0.08	12	0.28	59	0.19
alarm	37	35	0.95	81	0.27	1	0.0
coal_gasifier_risk	39	10	0.1	10	0.15	105	0.08
ecoli70	46	13	0.0	23	0.11	1	0.0
barley	48	11	0.0	10	0.1	2	0.02
hailfinder	56	14	0.0	20	0.11	2	0.0
hepar2	70	17	0.17	21	0.16	1	0.0
win95pts	76	17	0.16	21	0.09	26	0.04
munin1	186	15	0.0	16	0.03	1	0.0
neuro	222	10	0.0	19	0.01	1	0.0

Table 9: Full analysis of data contamination

Real nodes and their meaning *JSON of nodes and their corresponding meaning*

The node and their meaning LLM returned in the previous message *JSON of nodes and their corresponding meaning*

Compare the nodes and their meaning in Bayesian Network LLM returned in the previous with the real nodes. The nodes are considered to be similar even if the names slightly differs but their meaning is similar. Return the list of nodes that were returned in the previous message that also present in the real Bayesian Network.

Return JSON of form

"node from the real Bayesian Network": "node from the list you returned" (if they are similar)

Return empty JSON if no nodes are similar